Solving Problems in an Uncertain World

Solving Problems in an Uncertain World

A Thesis presented

by

Stuart Merrill Shieber

to

Applied Mathematics

in partial fulfillment of the honors requirements

for the degree of

Bachelor of Arts

Harvard College

Cambridge, Massachusetts

May 1, 1981

Note

This document is a digitized version of my senior thesis submitted to the Applied Mathematics concentration at Harvard College in Cambridge, Massachusetts on May 1, 1981 in partial fulfillment of the honors requirements for the degree of Bachelor of Arts. The approach and content is by now quite out of date, and is provided for historical reference as part of Harvard's repository collection of Faculty of Arts and Sciences theses and dissertations. (Of some slight interest might be Appendix D, a general critique of artificial intelligence research methodology, added, as I recall, at the request of one of my readers who was a strong AI skeptic.)

I recommend that readers use an accompanying version of the thesis available from the Harvard repository, which has been re-typeset using modern methods, with reworked figures replacing the original hand drawings and a few typographical errors corrected, though with no effort at systematically correcting errors.

For purposes of citation, the following form is recommended:

> Stuart M. Shieber. "Solving Problems in an Uncertain World." AB thesis, Harvard College, 1981.

Stuart M. Shieber
Cambridge, Massachusetts
July 17, 2013

Abstract

The issue of problem solving in the context of incomplete or inconsistent information is a precursor to designing multiple agent planning systems. We present some general principles of planning and problem-solving in uncertainty, and instantiate these principles in a problem-solving system based on the NOAH planning system. The new NOAH system, working on the blocks world as a test domain, plans in certainty with the same efficacy as the original system, but can also handle a large class of errors caused by inconsistent or incomplete information.

Table of Contents

Abstract
Table of Contents

1 Introduction
2 Scope of the thesis
3 An introduction to NOAH
4 General principles of planning in uncertainty
5 An overview of new NOAH
6 New NOAH and the blocks world
7 New NOAH plans in uncertainty
8 Assorted topics in planning in uncertainty with new NOAH
9 Summary

A Figures
B Substantiation of plan detail results
C Problems with the Resolve Conflicts critic
D A critique of Artificial Intelligence research methods
E Bibliography

1 Introduction

The ability to cooperate in solving problems is one of the capabilities that has distinguished humans as members of a social species and lead to their evolutionary success, capitalizing on the efficiency and synergistic advantages of group problem solving. In an effort to make computers more efficient, to allow them to take advantage of the same capabilities of distributed problem solving, such concepts as parallel processors and distributed data bases have arisen in computer science.

Artificial intelligence techniques have been derived for simulating intelligent solutions to certain classes of problems. Ideally, such techniques could be used as a basis for distributing the problem solving responsibility among several processors. This is the goal of the newly emerging topic of distributed artificial intelligence: (DAI) to endow multiple computing agents wih the ability to cooperate effectively in solving problems.

1.1 Distributed AI versus distributed processing

Distributed artificial intelligence differs from traditional distributed processing in several ways. Davis [4] has noted the following differences:

degree of control decentralization: Rather than a central computer controlling distributed processors doing very limited tasks, DAI attempts to model multiple agents acting independently but cooperatively.

conflict versus cooperation: Rather than concentrating on the problems of compromise among conflicting agents (e.g. access control, security, etc.), DAI views multiple agents in terms of benevolent problem solving activity.

Thus, DAI problem issues are different from, but bear some overlap with, distributed processing issues. In particular, some of the important problems of DAI include:

decomposition: the problem must be decomposed into sub-problems in an intelligent way

uncertainty: agents may have incomplete or inconsistent information

coherency: agents should not work at cross-purposes

communication: since communication is expensive relative to computing, cross-talk should be limited

2 Scope of the Thesis

No attempt will be made here at solving such a large problem. To limit the scope of this thesis, a particular domain was chosen and a subset of the issues researched with reference to this, admittedly limited, domain.

2.1 The blocks world

The domain investigated is the classical AI domain of problem solving in the blocks world. The blocks world consists of a table, some blocks, and agents that are robots with one hand and one eye. Each agent has a knowledge base, or world view, consisting of a set of assertions about the

state of the world.[1] Problems in the blocks world typically take the form of building structures out of the blocks.

Use of the blocks world as an area of problem solving in AI reaches back to the early vision research of Guzman-Averas [9] and the pioneering natural language understanding research of Terry Winograd [23] and has been consistently studied since then by innumerable researchers. It serves as a common paradigm for much of the research in problem solution planning on which this thesis is based. Although Winograd's research considered primarily language understanding, much work has been done on the planning aspect of problem solving in the blocks world. Fahlman [7] for instance, has demonstrated that solutions to quite subtle problems can be generated automatically by sophisticated enough systems. However, it is not the goal of this thesis to expand the set of blocks world problems for which computer solutions exist. Rather, it is to investigate standard existing planning methods for blocks world problem solving, and their distribution to multiple agents.

2.2 The NOAH planning system

---

1. Assertions typically are written as LISP-like prefix expressions in lower case, e.g., (cleartop A) or (on S D). Parenthesized prefix expressions written in upper case conventionally denote nodes in NOAH procedural nets rather than assertions, e.g., (CLEAR A) or (ON S D). Assertions can be combined with the standard logical and quantificational operators, although nowhere in this thesis is more than one level of quantification needed.

Planning systems, starting with the early GPS system [6], have steadily increased in sophistication and capability, through a variety of techniques -- means-ends analysis (STRIPS [8]), use of abstraction (ABSTRIPS [19]), nonlinearity of plans and hierarchical planning (NOAH [18]). The last of these, NOAH, is particularly suited to a study of distributed problem-solving, as will become apparent shortly. Corkill [3] used NOAH in his investigation of distributed problem solving for just this reason. We also use NOAH as the starting point on which to base a planning system for use in distributed problem solving.

2.3 The uncertainty problem

To constrain the problem further, we will examine in greatest detail the issue of uncertainty in an agent's planning. This issue was chosen for two reasons. First, it is readily isolable, since planning in an uncertain world need not be necessitated only by the existence of other problem solving agents. Second, it is a necessary precursor to distributed problem solving, since in the case other agents do exist, their independent actions can cause inconsistency and incompleteness in the world view of any given agent.

To summarize, then, the problem we attempt to solve concerns the solution of blocks world problems by an agent whose model of the world, or world view, may be inconsistent or incomplete. Without the context of a particular multiple-agent communication protocol, such issues as

concurrency, communication and problem decomposition cannot be discussed. However, preliminary work on a multiple-agent system using the new NOAH problem-solving system presented is presently being researched with just these issues in mind.

2.4 A disclaimer

The problem of uncertainty problem solving in the blocks world is, admittedly, a limited one, chosen for its tractability. It is our hope, though we make no claims, that the methods and principles discussed for expansion of NOAH to handle this particular problem will be applicable to other domains and problem issues in distributed AI.

3 An introduction to NOAH

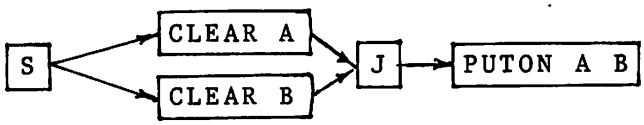
The problem solving system presented here is based on the NOAH system designed and implemented by Sacerdoti [18]. NOAH, like most existing planning systems, solves problems by planning actions, and then executing the plan as a guide for the behavior of the agent. Thus, the problem solution is a two phase process: planning followed by execution.[1] The planning phase consists of generation of incrementally more detailed procedural nets. The execution phase consists of performing the primitive actions implicit in the highest detailed procedural net.

---

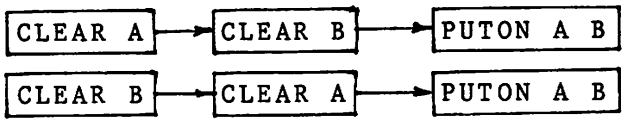
1. NOAH per se is only the planning portion of this problem solving paradigm.

3.1 The structure of the plans: procedural nets

Plans are structured as procedural nets which are networks of nodes each denoting an action. The partial order of nodes that the network imposes corresponds to a partial ordering of the actions the nodes represent. For instance, a plan for putting block A on top of block B might look like this:[1]



This plan can be thought of as shorthand for a set of linear plans that would achieve the assertion (on A B), viz.:



Thus, the parallel branches express the fact that any ordering of execution is acceptable, or even, that parallel execution would yield correct results. This, then, is the benefit of NOAH's use for distributed planning: that plans are expressed as potentially parallel operations whenever possible. The rationale behind allowing parallel branches in plans is to put off decisions as to the ordering of

---

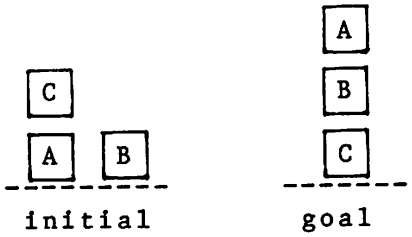
1. The split (S) and join (J) nodes are merely notational convenience for expressing the forming and coalescing of branches. The split node is used in different ways in new NOAH as will be discussed later.

actions as long as possible. This principle of "keeping your options open" is used extensively in new NOAH's extensions as well.

When a node of the procedural net corresponds to an action whose post-condition exists in the world, the node is marked as a phantom node, that is, a node that need not be expanded futher since its job is in some sense already done. These phantom nodes are denoted by rounded boxes.

3.2 The structure of behavior: NOAH's control structure

NOAH plans by generating, via a standard control structure, hierarchies of these procedural nets. Suppose NOAH were working on the following problem:



Starting with a one-node plan:

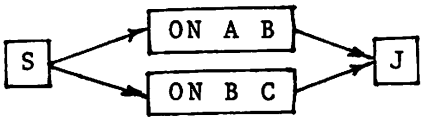
```
achieve ( and
  (on A B)
  (on B C) )
```

NOAH repeatedly expands and criticizes, until an executable plan (one consisting entirely of primitive actions) is generated, at which time, planning is complete.[2]

---

2. Special cases can arise when further criticism occurs after partial execution. (Cf. [])

Expansion of a particular node in the plan is according to an expansion template.[3] For instance, the template for the expansion of (ON A B) is shown in the beginning of section 3.1. For the current example, NOAH would first expand the plan as follows:



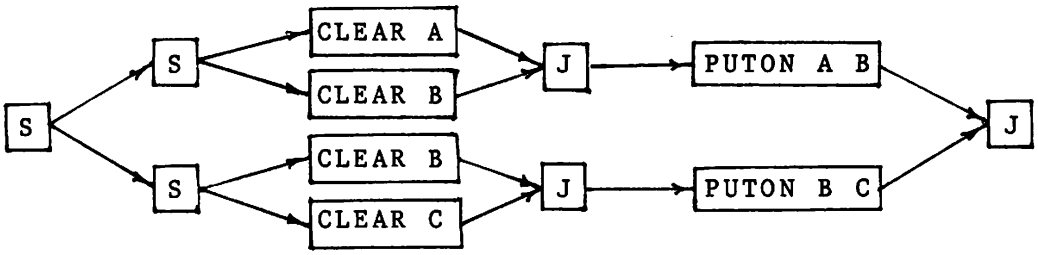
Criticism by the plan time critics involves no reordering of the nodes.

Note that NOAH leaves the plan in a nonlinear form. Committing to either linearization of the plan at this stage would not allow formation of a correct plan. Putting A on B first would merely make the moving of B more difficult. Alternatively, putting B on C just covers block A further. Either method leads nowhere. Thus, the parallelism implicit in NOAH's procedural nets is crucial to correct planning.

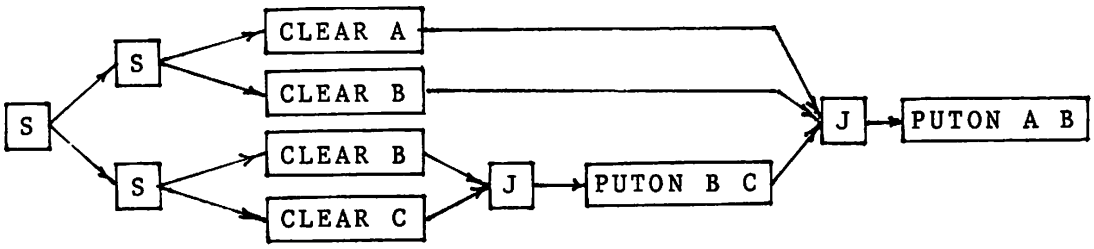
Further expansion yields:

---

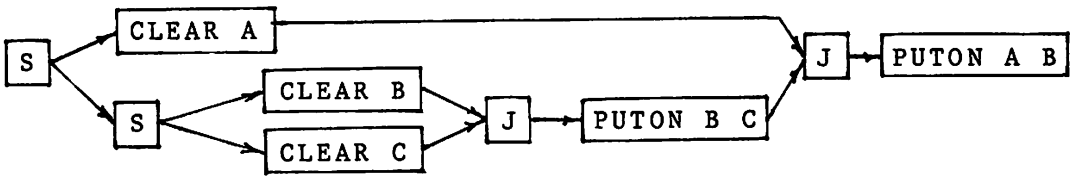
3. Actually, the expansion in [Sacerdoti] is according to information stored in a set of procedures written in SOUP code. This paper however uses the isomorphic method of procedural net expansion templates.



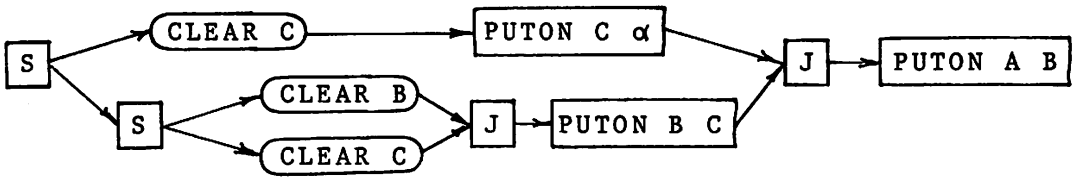
Now, the Resolve Conflicts critic -- one of the several critics that check plans during the planning phase -- notes that the lower (CLEAR B) node asserts a statement that the (PUTON A B) node denies, namely, the assertion (cleartop B). This is discovered through the generation of a table of multiple entries (TOME) based on standard assertions and denials associated with each type of node. The Resolve Conflicts critic reorders the plan to eliminate the conflict.



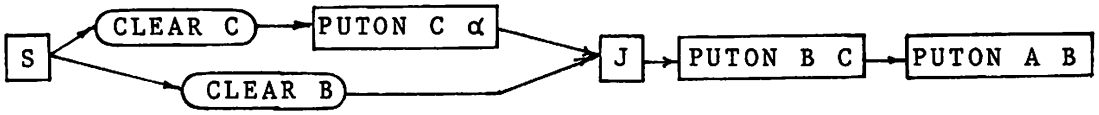
After further criticism, specifically, by the Eliminate Redundant Preconditions critic, we get:



The cycle of expansion and criticism is repeated. Since the only nonprimitive node in the plan is (CLEAR A), the new plan is expanded to:



Criticism by Resolve Conflicts and Eliminate Redundant preconditions yields:



This is the final expansion; the plan is almost completely linearized and execution of this plan would result in the moving of C to some empty space (α), then putting B on C, and finally, adding A to the top of this tower.

This simple example demonstrates the skeletal control structure of NOAH: repeated expansion and criticism of procedural nets. Sacerdoti includes several critics for plans -- Resolve Conflicts, Eliminate Redundant Preconditions, Use Existing Objects, Resolve Double Crosses, Optimize Disjuncts -- only two of which were used in the previous example. The technical details of the operations of the critics are presented in [18] and are therefore not discussed here. The new NOAH system makes use of them, and feasability of their use in the new system is based on their existence in the implemented version of NOAH.

4 General principles of planning in uncertainty

Having introduced the particular domain in which prob-

lem solving in uncertainty will be analyzed, we turn now to a discussion of some general assumptions and principles of planning in an uncertain world. We then describe the particulars of a solution in the test domain.

4.1 Assumptions of planning in uncertainty

Assumption 1: Although errors in an agent's world view can occur, they are in actuality relatively rare occurrences.

This assumption is certainly true in the real world. Intuitive evidence for the assumption's validity is given by imagining a world in which this assumption were false. Living in such a world -- where more often than not, objects change location while one's back is turned -- would be humorous at best, fatal at worst.

Assumption 2: Errors are assumed equally likely to occur at any point in the execution of a plan.

Errors are viewed as being caused by forces independent of the problem-solving agent -- by other agents or natural forces, for instance. This assumption may be inaccurate for domains in which certain actions are more prone to errors than others, but we choose to neglect this consideration.

4.2 Principles of planning in uncertainty

Principle 1: Execution should be monitored

Uncertainty, by definition, is the inconsistency or

incompleteness of an agent's world view relative to the real world in which it operates. To solve problems effectively in the real world, then, some checking must be done of the state of the world so that consistency of the world view can be forced. Since uncertainties are inherently asynchronous during execution, the checking should be done during execution. Clearly, this models the way in which humans solve problems, checking as they go that the world is as they think it is and that their actions are moving the world toward their goal.

Principle 2: Planning and execution should be interleaved

Now, assuming checking steps are inserted at strategic points in the plan, the results of such checks should effect the planning of later actions. That is, checks done during execution may proceed planning during problem solution. The result is that planning and execution are interleaved. NOAH, of course, with its strict two-phase control structure, does not implement this principle.[1]

Other reasons for interleaving planning and execution can be seen by examining alternatives. If planning strictly proceeds execution (as in NOAH), then the plan derived must contain contingencies for all uncertainties that the agent

---

1. Although some reordering of nodes during execution was planned by Sacerdoti in [18], execution time criticism was not implemented by the time of publication. Certainly, execution time criticism to the degree used in new NOAH was not envisioned.

wants to handle. Corkill [3] implements a multi-agent planning system based on NOAH that handles uncertainty without interleaving of planning and execution. However, to do this, Corkill implicitly assumes first that world view errors come about only by the actions of cooperating agents, and second that agents communicate enough information at plan time to catch all possible errors that might arise through concurrent execution of their plans, even though many of these errors might not come up in a particular linearization of the mutual plans. Thus vast amounts of communication are done to prevent errors that might not even come up. By interleaving of planning and execution, this communication becomes unnecessary.

Principle 3: Planning should assume an accurate world view

This principle seems at first to deny the very premise of uncertainty in the world. The resolution of this apparent paradox is derived from assumption 1, that world view errors are rare. Thus, a planning system should assume an accurate world view when making individual decisions as to, for instance, which of two plans is more efficient. Nevertheless, the planner should have the flexibility to handle errors when they do occasionally occur.

Humans solve problems on a similar basis. A construction worker who puts a tool down will go to the place s/he put the tool (demonstrating confidence in his/her world view). Only when the tool is not found there will s/he

begin communication to ascertain the correct current location of the object.

Principle 4: Plan detail should vary with time

As execution of plans proceeds, errors in the agent's world view may be discovered that make the rest of the plan inappropriate for achieving the goal. A corollary of assumption 2 is that the farther in the partial order a node lies (i.e., the later it would be executed), the more the likelihood that it will need to be replanned. In general, then, it does not pay to expand distant nodes in the partial order to the same level of detail as the closer ones, since the computation time used in the expansion of these nodes will more likely be wasted. The assumption in this reasoning that the waste is significant -- either that the amount of computation wasted is large or that the elimination of the computation involves a decrease in inter-agent communication -- will be substantiated in section 5.4.

Principle 5: Advantage should be taken of fortuitous situations

In the case of planning in a certain world, perfect information allows systems to take advantage of, for instance, portions of the goal already achieved. In uncertainty, on the other hand, agents may be unaware of such situations, and might plan to achieve an existing goal. Thus, planning systems should have the flexibility to change their plans accordingly when they discover such fortuitous

situations.

Principle 6: Time between checks and use of checks should be minimized

Another effect of assumption 2 is that the expected number of errors in a given time span is proportional to the length of the time span. Thus, to minimize the number of errors encountered, an agent should minimize the time it allows between checking some assertion about the world (i.e., forcing its world view to be consistent with respect to the assertion) and the use of that check (i.e., acting on the assumption of that consistency). For example, it should minimize the time between checking that block B is clear and putting block A on B. In particular, the assumption is made that if the check immediately proceeds the use during execution, with no actions occurring in between, then no errors will occur in performing the action.

These principles are by no means exhaustive. They merely represent the kind of thinking made concrete in the design of the new NOAH for planning in uncertainty. The instantiation of these principles in a planning system is the subject of the next section.

5 An overview of new NOAH

We discuss a substantial set of modifications to the original NOAH planning system that implement the general principles presented in the preceding section.

5.1 Execution is monitored

An agent planning with new NOAH has a world view, a set of assertions about the world, which may or may not reflect the state of the real world. New NOAH allows the agent to check, at various points during planning and execution, the state of the world or the world view. NOAH also allowed querying of the agent's world view, but since no uncertainty was possible in the original NOAH scenario, there was no reason to check both, or even to differentiate between the two. In new NOAH, when a query is generated during planning or execution, the agent checks the state of the world if possible (i.e., if the agent's hand/eye is in the right place at the right time); otherwise, it checks its world view by running a deductive theorem prover. This process is hereafter referred to as checking the world/view. Of course, the agent keeps track of facts it discovers through querying the world and updates its world view appropriately.[1]

The use of a deductive theorem prover in planning systems is not new. Many early planning systems (see [2]) consisted primarily of a theorem prover acting on a world view augmented by an appropriate axiomatization of a domain. Current planning systems such as Appelt's [1] and

---

1. A consequence of updating the world view is the problem of truth maintenance discussed in [5]. Other researchers have studied solutions to the general problem. No attempt is made to solve it here.

Wilkins/Robinson's SIPE [22] use a theorem prover or other techniques for deducing new facts from the world view.

5.2 Planning and execution are interleaved

NOAH planned by repeatedly expanding and criticizing, then executing. New NOAH plans by expanding and criticizing once, then executing, and repeating the whole process. Thus planning and execution are interleaved. To be more precise, new NOAH repeatedly expands nodes until all nodes at the front of the net are primitive, applying critics to the new procedural nets after each expansion, and then executes all primitive nodes at the front of the procedural net.

5.3 Planning assumes an accurate world view

New NOAH associates a purpose with each type of node. The (ON A B) node, for instance, has the purpose (on A B), and the (CLEAR A) node has the purpose (cleartop A). On the assumption that the world view is accurate, new NOAH expands using a node's template only if the purpose of that node is not satisfied. This decision is made by querying the world/view as per section 5.1. If the purpose of the node is satisfied, it is trivially expanded to the primitive node (NULL) whose associated action is to do nothing.[2] Thus,

---

2. Note that this concept of conditional expansion is similar to, but not identical with, that of phantom nodes. In fact, this feature of the control structure replaces the phantom node feature of the procedural net.

there may be instances when new NOAH queries an inconsistent world view, decides that a node's purpose is satisfied by the world when it in fact is not, and therefore effectively stops planning to achieve that purpose. The agent will inevitably meet with a rude surprise later when it checks the real world and replanning will be necessary. (See section 6.3.) This is the price one pays for planning in an uncertain world.

5.4 Plan detail varies with time

Another corollary of assumption 2 is that the probability of replanning a given node increases with its distance in the future, and that the relation is exponential. (See appendix B.) This, then is the impetus for leaving later nodes less expanded, so that the nodes which are more likely to be replanned have undergone fewer (potentially wasted) expansions. New NOAH implements a decreasing level of expansion by expanding only the first nodes[3] in the procedural net. This technique has the advantage that The number of nodes in the procedural net is bounded linearly by its depth of expansion, and thus, the expected number of wasted expansions over a given time span is effectively proportional only to the length of the time span. Thus, wasted work is effectively constant over the whole plan, rather than exponentially increasing with time, as the traditional

---

3. By first nodes we mean the nodes which are potentially the first to be executed in the procedural net.

NOAH control structure would be.[4] (Again, see appendix B for formal treatment of these considerations.)

The choice of expanding only the first nodes in the plan was arbitrary, for convenience only. In general, expansion of the first m nodes of the plan also yields the exponential decay in expansion level. By changing the value of m, the control structure could in effect be tuned to provide an optimal cross between early error detection and wasted expansion avoidance.

5.5 An example: two blocks

With this much of new NOAH's control structure explicated, we can demonstrate a simple example of block planning. The planning system demonstrated is actually a cross between the old and new NOAH systems, since major portions of new NOAH have yet to be introduced. The control structure includes the previously described new NOAH constructs, while the expansion templates are those of old NOAH.

5.5.1 A discussion of the notation for examples

Since expansion and execution proceed basically as a depth first forming and traversal of the hierarchy of procedural nets (since all expansions and executions are done

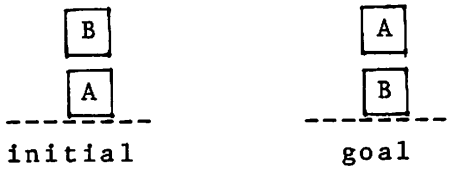
---

4. These arguments are substantiation for the intuition behind design decisions made in new NOAH. They are not presented as formal models of new NOAH's performance, since they are based on vast simplifications of the new NOAH planning system.

at the front of the net), we illustrate the dynamic action of NOAH on this and future examples as the static tree of procedural nets and actions that the new NOAH control structure forms (hereafter referred to as the action tree). The dynamics of the tree are retrieved by reading in a depth-first manner. The action tree for this example and the traditional notation are presented in figures 1.2 and 1.3 for comparison.

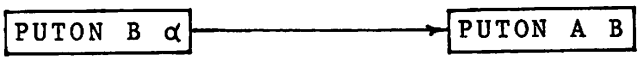
5.5.2 The problem and a solution

The problem is illustrated as follows:



B is on top of A initially. The goal state has A on top of B. The agent is presumed to have a complete and consistent world view. New NOAH begins with the single node plan (ON A B). This is expanded according to the template for the ON node as shown in the figure. (No reordering is done by the critics.) Now NOAH expands the first nodes in the plan, the two CLEAR nodes. Even if PUTON were not a primitive node (as it is in the traditional NOAH system), it would not be expanded, since it is not one of the first nodes in the tree. The (CLEAR B) node expands to (NULL) as its purpose (cleartop B) is asserted in the agent's world view. The (CLEAR A) node expands as per the old NOAH CLEAR template,

its purpose not being satisfied. (Again, no reordering is done by the critics.) Finally, the non-primitive node (CLEAR B) is expanded to (NULL) since it is first in the partial order. Now all first nodes in the plan are primitive. They are executed (in arbitrary order), the agent does nothing (twice) and planning continues. The plan now looks like:



Again, the first node in the plan is primitive so it can be executed. Were it not primitive, it would have been expanded until the first nodes were primitive. Finally, the goal is achieved by executing the (PUTON A B) node.

5.6 More execution monitoring: execution time critics

Execution monitoring through world/view querying was briefly explained in section 5.1. We have seen one way in which this occurs -- by the conditional expansion of nodes. A more general method of using world/view queries to guide execution is through the use of execution time critics in the plan itself.

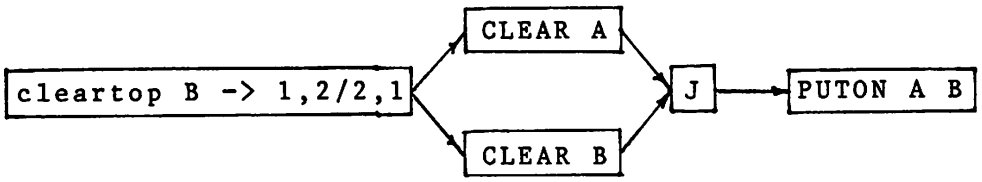
Execution time critics are essentially specialized split nodes. When these nodes are executed, they query the world/view and reorder their branches based on the outcome of the query. An execution time critic is of the form: (<assertion> -> <true list> / <false list>) where <assertion> is a theorem that the theorem prover will check at

execution time, and the true and false lists are ordered lists of numbered branches. On execution, if the assertion is true, the branches of the critic node are reordered as per the ordering in the true list; otherwise, the false list is used for reordering. Note that since the branch lists need not include all the branches, the execution time critic can act as a device for conditional execution as well as reordering.

An example of an application of execution time critics occurs in the expansion template for the (ON A B) node. Rather than choosing arbitrarily whether to (CLEAR A) or (CLEAR B) first, as in the two blocks example, we could choose based on the state of the world/view at execution time. For efficiency's sake (i.e., to minimize hand motions), an agent should prefer to put off clearing B if it thinks it is already clear, so that it can pick up A just after it has cleared it, thus saving one hand motion. On the other hand, if B is presumed to be covered, it would be best to clear it first, so that A can be picked up immediately after clearing A without having to first make a side trip to clear B. Ideally, then, the (ON A B) template would be:[5]

---

5. Unfortunately, the benefit of this approach does not appear until planning down to the level of hand motions is done. Such detailed planning is done by the set of blocks world templates presented in section 6.



Of course, the simple heuristic embodied in this expansion is not ideal in all cases. Although one hand movement is eliminated, in some instances (where block A is covered by many more blocks than B, for example) clearing A first might be advisable, so as to minimize the exposure time between B being cleared and A being placed on B (in accordance with principle 6). The critic method could still apply to more sophisticated heuristics although the simple form of the execution time critic presented here might have to be augmented to increase its expressive power.

The execution time critic allows error handling to be performed at execution time so that planning for all contingencies at plan time is unnecessary. Thus world/view checking can explicitly effect later planning and execution.

6 New NOAH and the blocks world

We sidestep for a moment to discuss the particular instantiation of new NOAH devised for problem solving in the blocks world. A set of templates is presented that, in conjunction with the new NOAH control structure, perform blocks world problem solving. These templates act in much the same way as Sacerdoti's SOUP code procedures except that planning is done down to the level of detail of individual hand

motions.[1] Thus, the primitive actions are GRASP, UNGRASP and MOVE, rather than the higher level action PUTON used as the primitive in [18]. This level of detail was chosen to show the varying detail with time implicit in the control structure and to allow some interesting optimizations using execution time criticism, such as the one presented in section 5.6.

6.1 The primitive nodes

The primitive nodes all perform actions relative to the location the world view associates with blocks -- the so-called world view location -- which may or may not correspond to the actual location. For instance, (GRASP X) grasps whatever block is at the world view location of X (assuming the robot hand/eye is at that location and the block is cleared). (MOVE X) moves the hand/eye to the world view location of X. (UNGRASP X) releases the block held so as to rest on top of the block at the world view location of X (again assuming the correct location of the hand/eye and the block cleared). The world view location is used so that these primitive actions are well-defined in the context of an uncertain world. The side effect of this mode of operation is, of course, that the primitive actions may not have the effect one would expect if the world view is incorrect and the world view location does not match the actual

---

1. The primitive actions at this level of detail are based loosely on those used in [23].

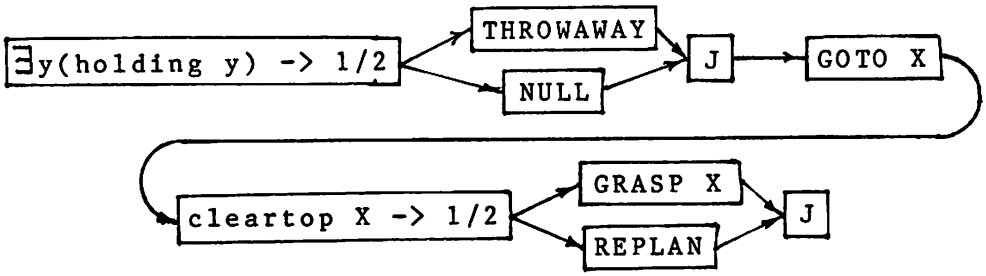
location. (MOVE X) for example may not move the hand/eye to block X. Thus, it cannot make the assertion (at x). (The GOTO node can however.) Checking must be done to assure that this goal has in fact been achieved, and appropriate action taken in case of failure. Much of the planning work done by new NOAH is just such contingency checking.

Other primitive nodes include the NULL node, whose associated action is to do nothing. The purpose, assertions, and denials are inherited from the generating node if the NULL node was derived as a default expansion. (See section 5.3.) The BIND node allows for explicit execution time world/view querying. Its format is (BIND X TO Y) where Y is a formal variable and X is an assertion free in Y. The REPLAN and FIND nodes will be explained in sections 7.3 and 7.5.

6.2 The higher level nodes

The full set of blocks world templates is shown in figure 2. Associated with each node is an expansion template, a purpose, and a set of assertions and denials that the node invokes. The template is used as an expansion for the node if the purpose of the node is not satisfied according to the world/view. The purpose of the node is used in this decision and in purpose-tracking (to be discussed in section 7.1). As before, the assertions and denials are used by the NOAH critics to detect conflicts that might require reordering of the procedural net.

The actions of the templates are relatively self-explanatory, as are the purpose, assertions and denials.[2] For example, the (GET X) node expands to:



which, informally translated, means "if holding anything, throw it away; then go to where X really is, and, if X is clear, grasp it, otherwise replan, correcting the assumption made earlier that X was clear."[3]

The blocks world templates presented in figure 2 and used throughout the rest of the paper are merely a sample set of planning templates. They demonstrate that blocks world planning in uncertainty is at least a feasible proposition. They are not a definitive set in any sense. For instance, it may be that with specific information as to the frequencies of certain error types, another set of templates

---

2. The only type of assertion occurring in the new NOAH templates that was not in old NOAH is the assertion (avail X). The significance of this assertion is that X is not in its final position, i.e. that it is available to be moved.

3. Note that we do not (CLEAR X), (GOTO X) and (GRASP X) after the empty-hand check. As explained in principle 6, correct operation of a GRASP node for instance is only guaranteed if the check for X clear occurs immediately prior to the grasp.

tends to be more efficient than the set presented. Such distinctions are, however, in general difficult to formalize in the face of such information, and are clearly moot without the information. These templates then are not how one should, but how one could solve blocks world problems in uncertainty.

6.3 Examples of new NOAH in action

Planning systems of the complexity of NOAH are in general extraordinarily difficult to prove correct.[4] With the addition of planning at a more detailed level, with execution monitoring and asynchronous error introduction, the problem is effectively, as well as technically, unsolvable. Consequently, through Sacerdoti's case analysis of blocks world problems, a set of canonical blocks world examples have been defined for informally demonstrating the minimal adequacy of block planning systems. These sample problems present the major difficulties a planning system will be called upon to solve in a wide class of blocks world problems. Although not a proof of correctness, they provide a strong basis for confidence in a system. The fact that new NOAH solves these problems, and solves them in substantially the same way as the traditional NOAH system, with no extra executed actions, founds a conviction that new NOAH plans in

---

4. Efforts of Rosenschein [17] to prove the correctness of even small portions of NOAH ended in the conclusion that the problem was at this point insurmountable.

a certain world at the same level of correctness as NOAH. Furthermore, solutions for a set of sample problems demonstrating the major categories of difficulties in planning with uncertainty add credence to new NOAH's ability to plan in uncertainty.

Problem solutions are presented in the action tree notation. The reordering done by critics is based on the reordering done by the original NOAH system in similar situations. Thus, the existence of an implemented NOAH is the argument for the feasability of these reorderings. The actions of the critics are not discussed fully in the context of these problems since they are adequately presented in [18].

6.3.1 Planning in certainty: the canonical set

6.3.1.1 Two blocks

We revisit the two blocks problem in the context of the full new NOAH control structure and the sample blocks world templates of section 6.2. (The action tree is figure 3.2.)

The single goal node (ON A B) is expanded as per the ON template. Executing the critic (cleartop B -> 1,2/2,1) causes reordering of the (CLEAR) nodes with A being cleared before B, since the critic noted (by querying the world/view) that B was clear. Now the (CLEAR A) node, the first node in the procedural net, is expanded. First, the (GOTO A) node further expands to a (MOVE A) and a check to

make sure that the hand/eye is actually at A. Since it is, the execution time critic reorders the second half of the GOTO expansion to yield a (NULL) node which is then (trivially) executed. The rest of the (CLEAR A) node expansion now heads the procedural net. The (cleartop A -> 1/2) critic is executed (noting that A is not clear) yielding the reordering to achieve removal of the offending block from A. The BIND nodes at the front of the procedural net are executed, associating the block B to be moved with the formal argument Y and the clear space α with Z. Now, new NOAH moves B from on top of A by expanding the (ON Y Z) node with the appropriate bindings.

The new NOAH control structure has now effectively recurred; trying to achieve (ON A B), it has generated the node (ON B α). Similar expansions occur for the (ON B α) node as occurred for the (ON A B) node. It is expanded and reordered. The two CLEAR nodes are expanded to (NULL) since their purposes are satisfied. The (PUTON B α) node is expanded. To (GET B), the agent makes sure it is not holding anything, and then heads to B, and grasps it since it is clear. The hand/eye is moved to the world view location of o and a check is made to ensure that the location is correct. Finally, after noting that o is indeed clear, the agent ungrasps B at α.

The situation now has both A and B resting on the table with the hand/eye at B. The (CLEAR A) node from the (ON A

B) expansion has been fully executed. The front node in the procedural net is (CLEAR B) which is expanded to (NULL) and executed.

Finally, the (PUTON A B) node is expanded and executed in a manner exactly analogous to the expansion and execution of (PUTON B α). Block A is grasped, moved to B, and ungrasped. The goal has been achieved.

The linearization of world view queries, world checks, and primitive hand actions generated by new NOAH is presented in figure 3.3. Note that new NOAH performed effectively the same actions as the traditional NOAH would have, though at a finer level of detail.

An important feature of the execution of this plan was that every actual hand movement was immediately preceded by a world check guaranteeing that the hand movement is executed correctly. For instance, every GRASP action is preceded by a real world check that the block to be grasped is clear. This feature, minimally necessary (though not sufficient) for the correct operation of plans, is implicit in the design of the blocks world templates presented, and is true in general for every hand motion executed.

Just because the individual hand movements are correct does not mean that the sequential execution of movements achieves the goal, however. Conflicts can arise that could endanger the correctness of the plan. It is the plan time

critics which catch such global errors and it is the correct action of these critics which is demonstrated by the remainder of Sacerdoti's canonical set.

The next few examples of blocks world problem solving demonstrate the major conflict situations which arise in blocks problems. We ignore the expansion of the (GOTO) and (THROWAWAY) nodes in presenting the action trees for these examples. The (GOTO) node is treated as equivalent to the primitive node (MOVE) since in a certain world, (MOVE X) always results in the assertion (at X) being true, and thus, the remainder of the (GOTO) exapnsion will always expand to (NULL). (THROWAWAY) is never used in these examples. Furthermore, once new NOAH has planned and criticized achieving a plan linear enough to resolve the conflict, we ignore further execution, since correctness is clear. Often, we will ignore the expansion and execution of obvious branches of the tree and note only what that branch achieves. These conventions are used to maintain the brevity and clarity of the samples without sacrificing correctness.

6.3.1.2 Three blocks

The problem solved by traditional NOAH in section 3 is presented as a problem to new NOAH. (See figure 4.1.) The conjunctive goal (and (ON A B) (ON B C)) is expanded as two parallel goals. Each of these ON nodes is further expanded and the plan time critics are applied. As in the

traditional NOAH scenario, the Resolve Conflicts critic builds a TOME and spots conflicts requiring reordering. After reordering is completed, the execution time critics are invoked and the almost completely linearized plan shown as the final procedural net in figure 4.2 is derived. The (CLEAR) nodes will all be expanded trivially, except for the (CLEAR A) node which will involve moving C to a clear spot. Then B can be placed on C and A on B. The conflict is resolved correctly by new NOAH.

6.3.1.3 Four blocks

Sacerdoti's four blocks problem is presented in figure 5. The Resolve Conflicts and Use Existing Objects critics are demonstrated in this example. It should be noted that the solution obtained is identical in practice to the traditional solution.

6.3.1.4 Creative Destruction

In this example (figure 6), a subgoal of the primary goal is already achieved in the initial world. Achievement of the primary goal involves temporarily dismantling the existing subgoal. NOAH recognized this by converting a phantom node to a normal goal node. New NOAH performs the isomorphic action of converting a NULL node back to the node that generated it and reexpanding the node. The realization that such action is necessary is brought about by the Resolve Conflicts critic in both NOAH and new NOAH.

6.3.1.5 Swapping Blocks

By introducing a new type of node, the (ABOVE) node (see figure 2), a new type of conflict can be generated. NOAH detects a double cross between the two (PUTON) nodes much as new NOAH does and reorders the plan using information based on its analysis of binding errors. (See figure 7.) Because the binding is done explicitly in new NOAH, the error detection is even more straightforward. The Resolve Double Cross critic acts just as in traditional NOAH generating a viable plan.

6.3.1.6 Disjunctive goals

Work with disjunctive goals in new NOAH has not been completed. It seems clear, however, that extending new NOAH isomorphically to NOAH in this context should be sufficient to implement disjunctive goal planning as well.

The full action of new NOAH on the canonical set examples is presented in figures 3 to 7. New NOAH finds solutions identical in form to those found by NOAH, although in much more detail. In addition, the plans generated are adequate for catching errors in the world view and acting accordingly. The uncertainty examples presented in the next section demonstrate this capability.

7 New NOAH plans in uncertainty

New NOAH uses a variety of methods to implement plan-

ning in the context of an errored world view. These methods are described in general and in the context of examples to demonstrate new NOAH's action in a variety of error situations.

Inconsistencies can be discovered in three basic ways during execution. The first two involve discovering in the real world the negation of a single variable assertion in the world view. Errors of this form can be either fortuitous or disadvantageous -- fortuitous in that they may remove the need to perform actions being planned, and disadvantageous in that they may require the changing of plans previously formed. (There is, of course, the possibility that the error does not affect the plan one way or the other, which case is handled trivially by just updating the world view appropriately.)

The final type of error occurs when the negation of the world view assertion results in a decrease in the amount of information in the view. Single variable assertions (assertions about the characteristics of single entities) have the property that either the assertion or its negation carry the same amount of information about the entities involved. (For instance, (cleartop A) and ~(cleartop A) reveal the same amount of information about A's state.) Multi-variable assertions (assertions about the relationships of entities) do not have this property. Thus, discovering the negation of a multi-variate assertion leads to a decrease in the com-

pleteness of the world view. This type of world view error is more insidious than the previous two precisely because of the incompleteness problem. New NOAH has devices to handle all of these types of inconsistencies.

7.1 Purpose-tracking allows taking advantage of fortuitous situations

New NOAH associates a purpose with each node in the hierarchy of procedural nets. We have seen these purposes used in deciding locally whether or not a node should be fully expanded. If an error in the world view leads to such expansion when it is actually unnecessary, and this error is discovered later, the hierarchy of purposes can be used to globally inhibit further expansion.[1] By constantly checking the net to see if the purpose of an ancestor node (a node in the hierarchy that eventually generated the current node) is satisfied, new NOAH can stop work on all nodes below the fulfilled node in the hierarchy.[2]

7.2 An example: missing blocks

In the blocks world domain, we can see this type of error in the following scenario. An agent is given the goal

---

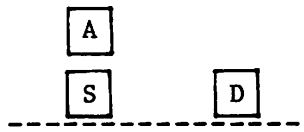
1. NOAH performs similar checking of nodes in the hierarchy (the so-called "hierarchical kernel") to ensure that goals of the higher nodes are being satisfied as planning progresses.

2. Clearly if more than one ancestor node has its purpose fulfilled, the higher in the tree should be inhibited since it subsumes more nodes underneath it.

of achieving (on S D) in an initial world:

initial world

but his world view is:



initial world view

In figure 8.2, the tree for this example is shown. Note that while expanding the node (CLEAR S), the agent discovers that S is clear thus satisfying the purpose of the CLEAR node, namely (cleartop S). Therefore, the agent stops work on the expansion of the (CLEAR S) node, acting as if the node had been fully executed.

Figure 9 shows a similar example, in which the destination block D rather than the source block is discovered to be clear.

Implementation of purpose-tracking is posited as a parallel process, a demon [24] or sprite [12] type of construct. Such parallel constructions have been used with success in other DAI systems, particularly distributed Hearsay II [14].

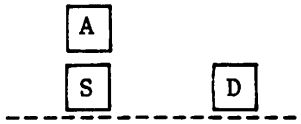
7.3 Replanning allows handling of previous plan errors

When new NOAH comes across situations in which a strong conflict between the existing plan and the real world exists, some inconsistency in the world view must be to blame. New NOAH is signalled that such a situation exists by the execution of a REPLAN node. The REPLAN node takes the last assertion made (typically due to checking of the world by an execution time critic or a purpose check) and examines its ancestor nodes for a node that utilized the negation of the assertion in question. The offending node's parent is then replanned.

REPLAN nodes are inserted in the blocks world templates before the GRASP and UNGRASP primitives, because these actions have preconditions which are absolutely inviolable, i.e. the actions will fail without satisfaction of the preconditions. Since these nodes are only reached at a time when the real world can be checked for the nodes preconditions, replanning can be appropriately invoked.

7.4 An example: extraneous blocks

The converse to the previous example is a situation in which the world view of the planning agent notes no covering block when one actually exists. For instance, an initial world:



initial world

and an initial world view:

initial world view

yield errors on attempting to achieve (on S D). Figure 10.2 shows new NOAH planning this example. It plans on the basis of the world view assertion (cleartop S) which is invoked by the node (CLEAR S) when new NOAH is deciding whether to expand the CLEAR node. Thus, when the REPLAN node is executed, (CLEAR S) is pinpointed as the offending node, and its parent (ON S D) is replanned.

Similarly, in figure 11, an example is provided with the destination block, rather than the source block, covered. Here, the critic (cleartop D -> 1,2/2,1) is the offending node and the (ON S D) node is replanned. In both examples, the correction of the world view allows the goal to be eventually achieved.

7.5 Errors causing incomplete information are handled differently

If an error is discovered that results in incomplete rather than inconsistent information, a completely different approach is necessary for resolution. Unless some other

agent has the appropriate information, such errors are fatal. Thus, solution of this problem is intrinsically tied to assumptions about what other agents exist and how agents interact with each other.

In the blocks world, errors of this sort occur when an agent discovers it does not know the correct location of a block. No amount of replanning can remedy this situation. Thus, a new primitive node, the FIND node, is introduced whose sole purpose is to gain, through communication with other agents, consistent information about block locations.

The actual method used in executing a FIND node is, as previously mentioned, dependent on the communications protocol implicit in the multiple agent system. For purposes of the example, the FIND node can be thought of as requesting information from some oracle of truth. Within the context of a particular multiple agent planning system, we have done some work on actually implementing a FIND algorithm. At worst, the agent could simply broadcast a request for information to all other agents and wait for a reply. Other more efficient methods are being examined.

7.6 An example: finding a block

Figure 12 shows a new NOAH planning tree for a trivial example utilizing the FIND node. Note that the GOTO node recurs in case the FIND node yielded inconsistent information.

8 Assorted topics in planning in uncertainty with new NOAH

8.1 Some blocks world errors are not handled

Certain errors were deemed of too severe a character to be handled by these simple methods. Specifically, the agent is assumed to have accurate information about its own state, i.e., its location and what it is grasping. Thus, the (at x) and (holding x) assertions in an agents world view are assumed always accurate. Without these minimal conditions, error checking becomes complicated by the lack of a reference of truth. These constraints seem reasonable, because the errors that should be handled are those caused by the actions of other agents.

8.2 The class of blocks world errors handled is large

Most blocks world errors, on the basis of case analysis, fall into one of the following categories discussed previously:

1. covered block thought clear
2. clear block thought covered
3. block location inconsistent

The first of these categories is handled by REPLAN, the second, by purpose-tracking, and the last category, by the FIND node. Thus a large number of blocks world errors can be handled by the system.

8.3 Indefinite postponement is possible

Clearly in the face of appropriately insidious errors, replanning could continue indefinitely. This problem was not considered serious for two reasons: First, assumption 1 makes long postponement quite unlikely. Second, the possibility of indefinite postponement exists in the real world as well as the limited blocks world domain. Anyone who has been beaten to several parking spaces in a row will vouch for its existence.

8.4 Irreversible actions require special treatment

The blocks world primitives are reversible operators. Grasping can be undone by ungrasping. Moving can be undone by moving back. Many domains allow primitive actions that cannot be reversed, however. In these domains, interleaving of planning and execution can be a dangerous proposition, since replanning may not always be possible. In such cases, special efforts -- such as complete expansion of nodes associated with irreversible actions, postponing execution, or increasing the number of first nodes expanded (the m of section 5.4) -- may be desirable to reduce the incidence of irreversible errors.

8.5 Planning with a highly errored world view

Using world views which are extremely inconsistent certainly leads to inefficient planning and execution. Much time is spent in information gathering and replanning. Worse than inefficiency, however, is the possibility that

goals may not even be achieved in the case where the world view initially claims (erroneously) that portions of the goal have been achieved. In the worst case, an agent can be given a goal whose purpose is satisfied by its world view but not by the world. No execution will ever be done by the agent, even though the goal is not achieved.

Thus a minimum level of correctness on the part of the world view is necessary so that the agent can at least identify that its goals are not achieved. Assumption 1 provides a basis for the reasonableness of this requirement.

9 Summary

The goal of cooperative problem-solving by multiple agents has been investigated by a number of researchers ([3], [10], [11], [13], [14], [20], [21]). One of the underlying issues all such research must address is the problem of planning in a world in which one's information may be incomplete or inconsistent. Assuming that errors in one's world view are rare and occur asynchronously of the agent's actions, we can identify several principles that should be embodied in execution monitoring (as opposed to probabilistic) problem-solving systems that work in uncertain worlds such as a worlds with multiple agents:

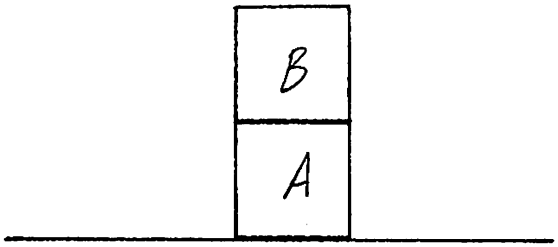
1. Execution should be monitored
2. Planning and execution should be interleaved
3. Planning should assume an accurate world view
4. Plan detail should vary with time
5. Advantage should be taken of fortuitous situations
6. Time between checks and use of checks should be minimized

The new NOAH system presented in this thesis is an embodiment of these principles. The control structure, with its execution time critics and conditional expansion, allow execution monitoring and interleaving of planning and execution. The early node expansion method varies plan detail with time. World/view querying assumes an accurate world view but allows error handling. Purpose tracking takes advantage of fortuitous situations. Finally, the design of the blocks world templates incorporates minimization of exposure time between world/view checks and their use.

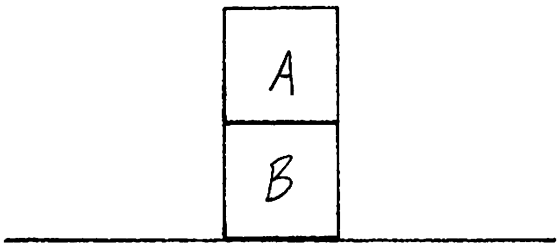
The new NOAH system applied to the blocks world passes several tests of usefulness. Sacerdoti's canonical set of blocks world problems demonstrate a minimal proficiency of planning in certainty with no apparent loss of execution efficiency relative to NOAH. A set of examples demonstrating the major categories of blocks world errors indicate the utility of new NOAH in an uncertain environment.

Planning in uncertainty is a necessary precursor to multiple-agent problem solving. New NOAH's evincement of the feasability of such planning and the explication of the principles involved may provide aid in this first step toward cooperation among problem-solving robots.

Appendix A: Figures



initial



goal

Figure 1.1: The two blocks problem

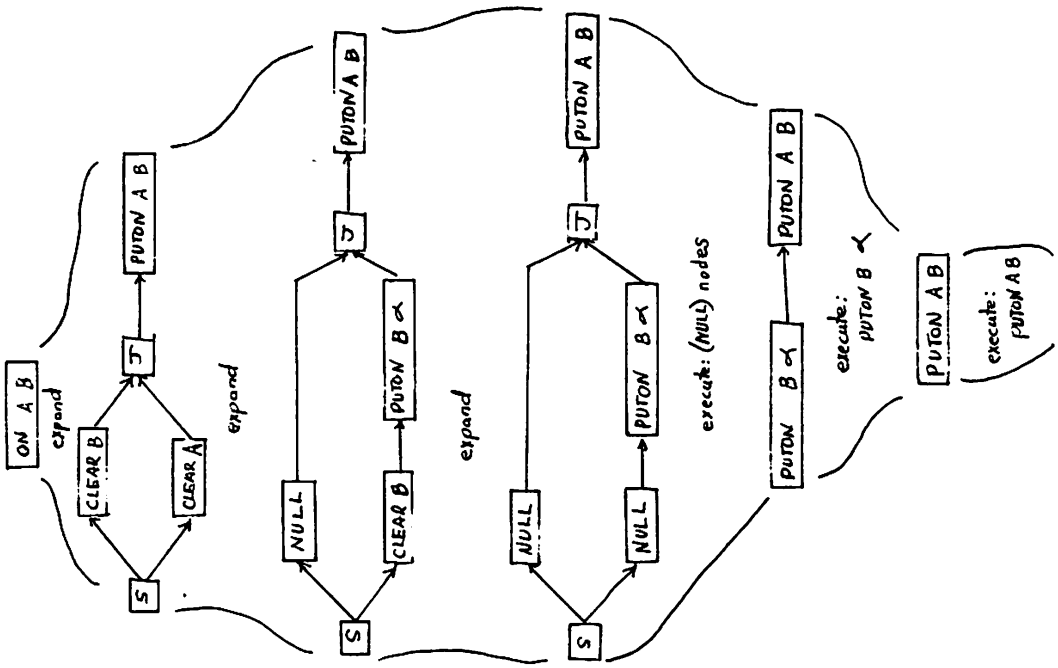


Figure 1.3: Solution to two blocks problem -- standard notation

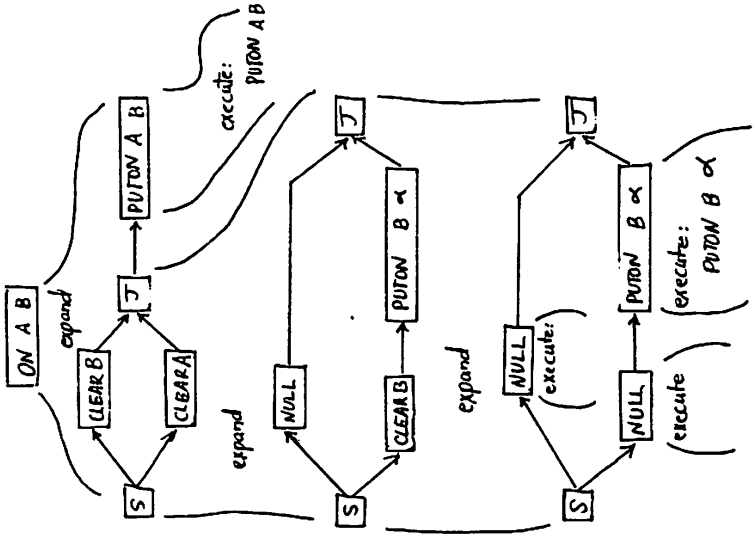


Figure 1.2: Solution to two blocks problem -- action tree notation

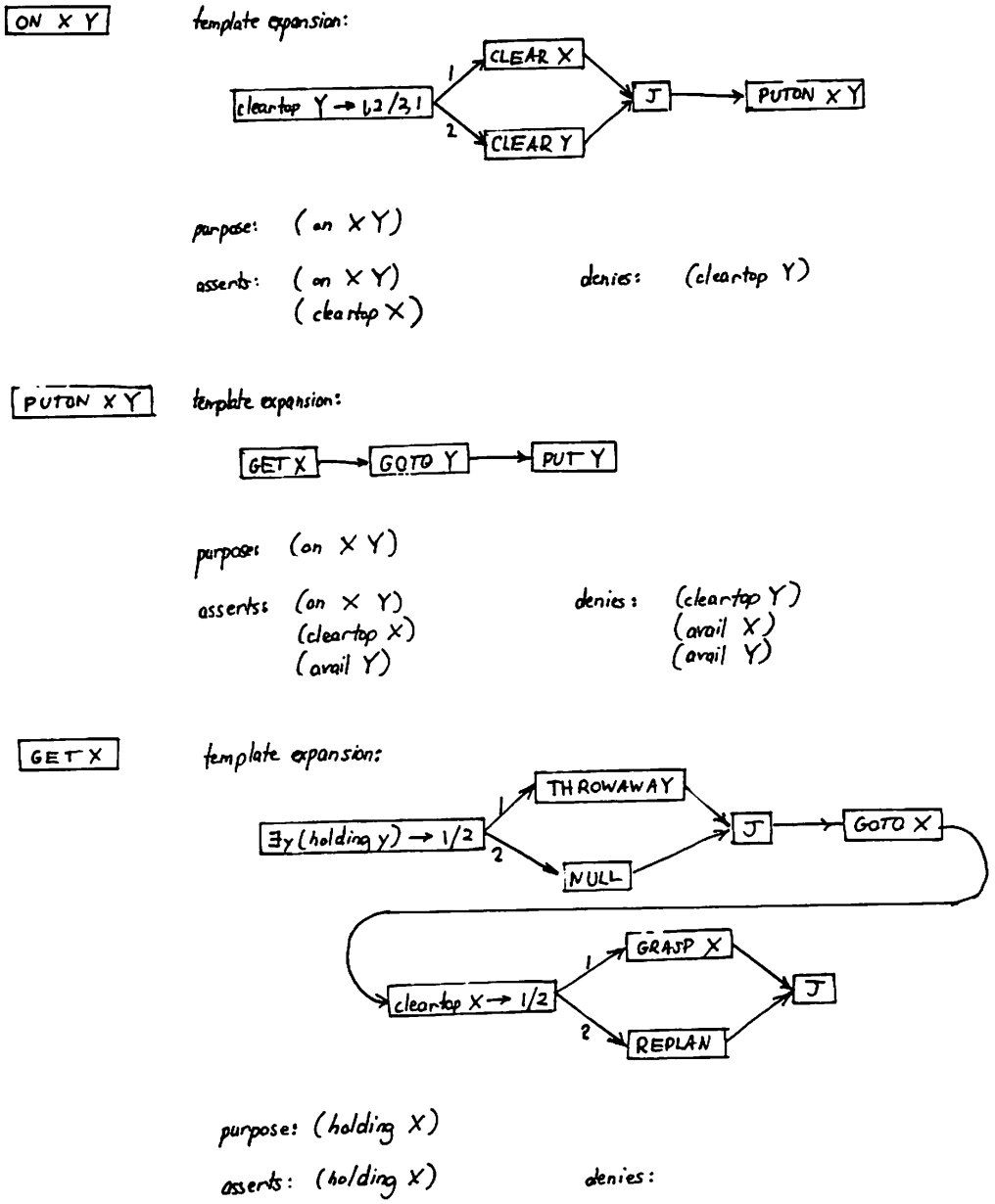


Figure 2: The blocks world templates for new NOAH

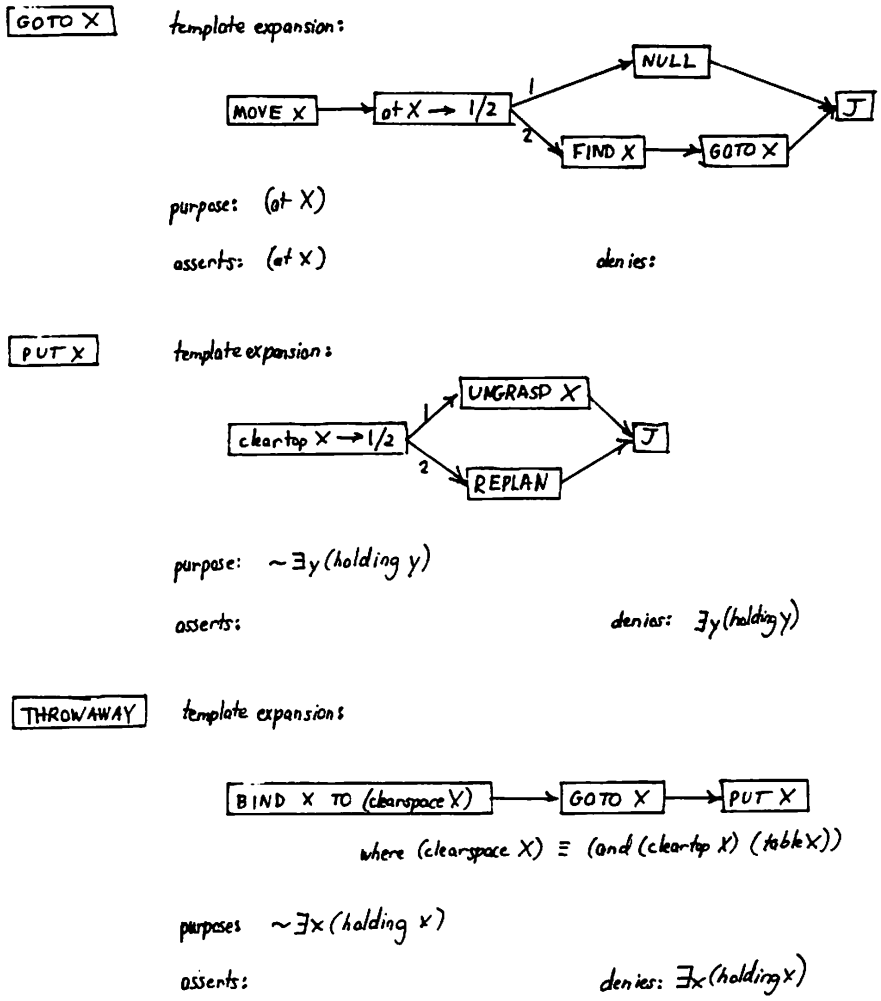
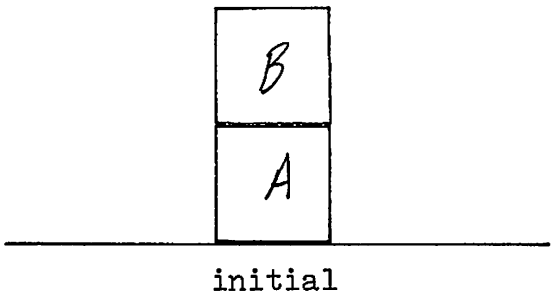


Figure 2: (continued)

Figure 2: (continued)



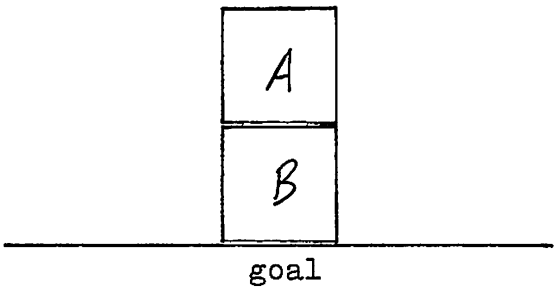


Figure 3,1: The two blocks problem

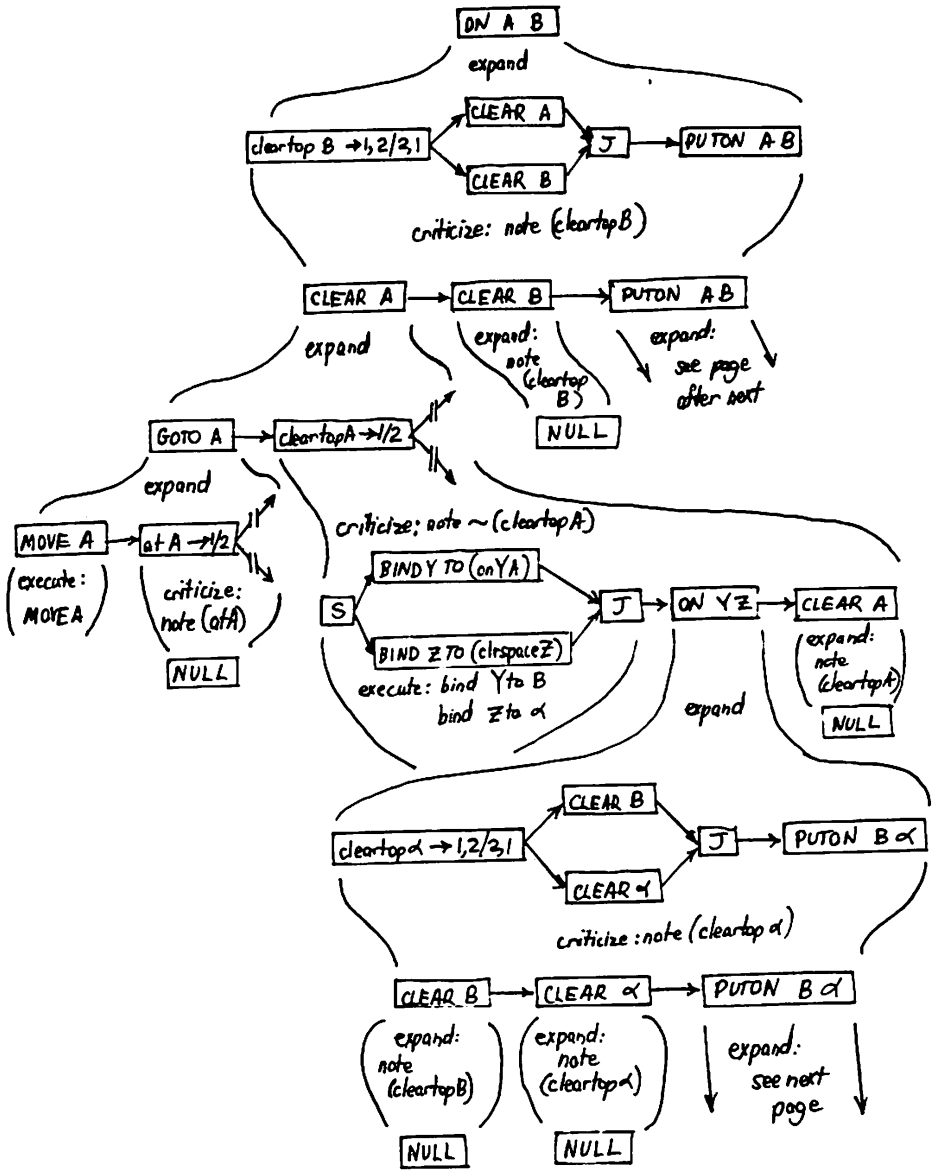


Figure 3.2: Solution for the two blocks problem

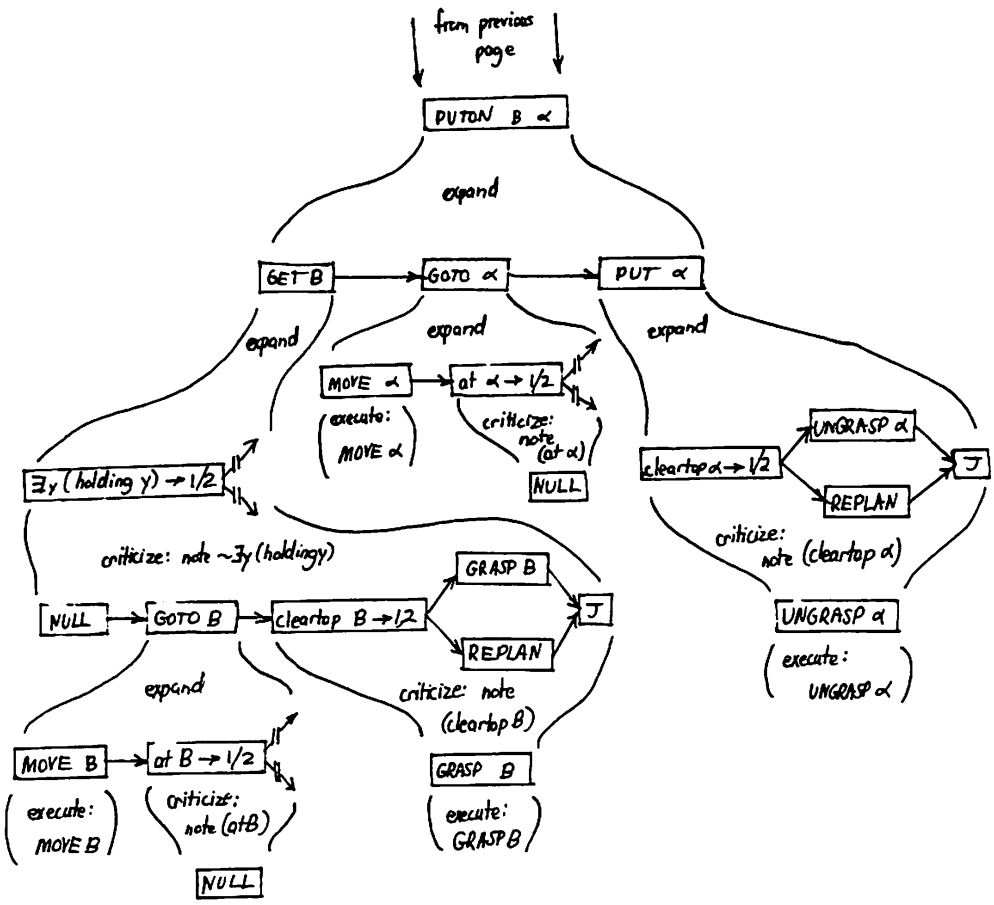


Figure 3.2: (continued)

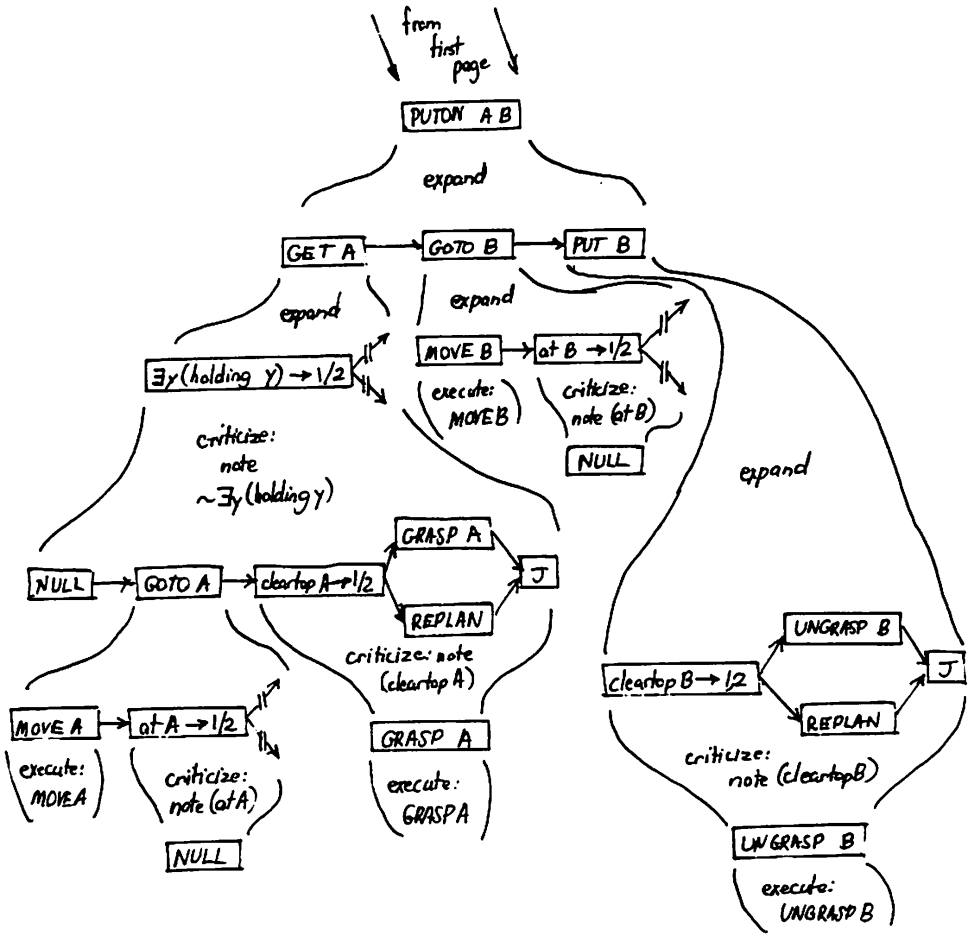


Figure 3.2: (continued)

world view queries:	world checks:	actions:
(cleartop B)		
		(MOVE A)
	(at A)	
	(cleartop A)	
(on B A)		
(cleartop α)		
(cleartop α)		
(cleartop B)		
(cleartop α)		
	(holding nothing)	
		(MOVE B)
	(at B)	
	(cleartop B)	
		(GRASP B)
		(MOVE α)
	(at α)	
	(cleartop α)	
		(UNGRASP α)
(cleartop A)		
(cleartop B)		
	(holding nothing)	
		(MOVE A)
	(at A)	
	(cleartop A)	
		(GRASP A)
		(MOVE B)
	(at B)	
	(cleartop B)	
		(UNGRASP B)

Figure 3.3: Linear ordering of the two blocks solution

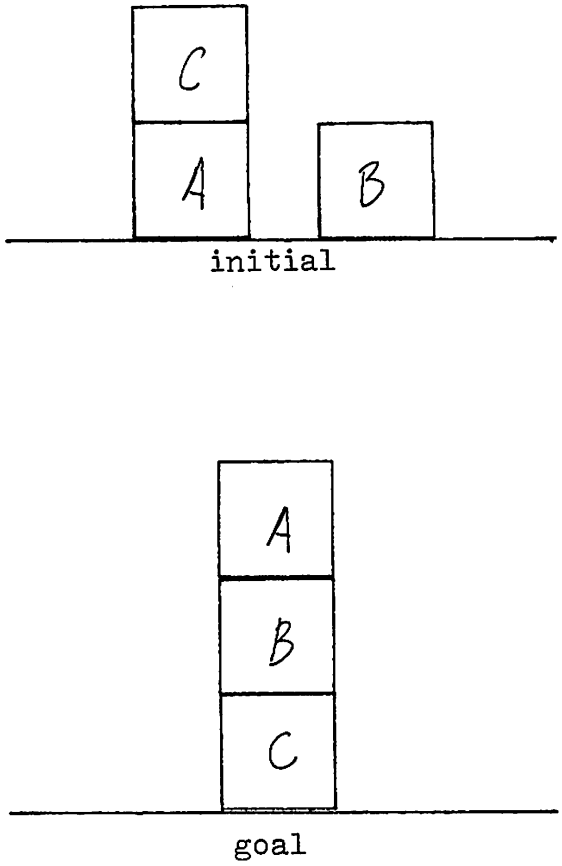


Figure 4.1: The three blocks problem

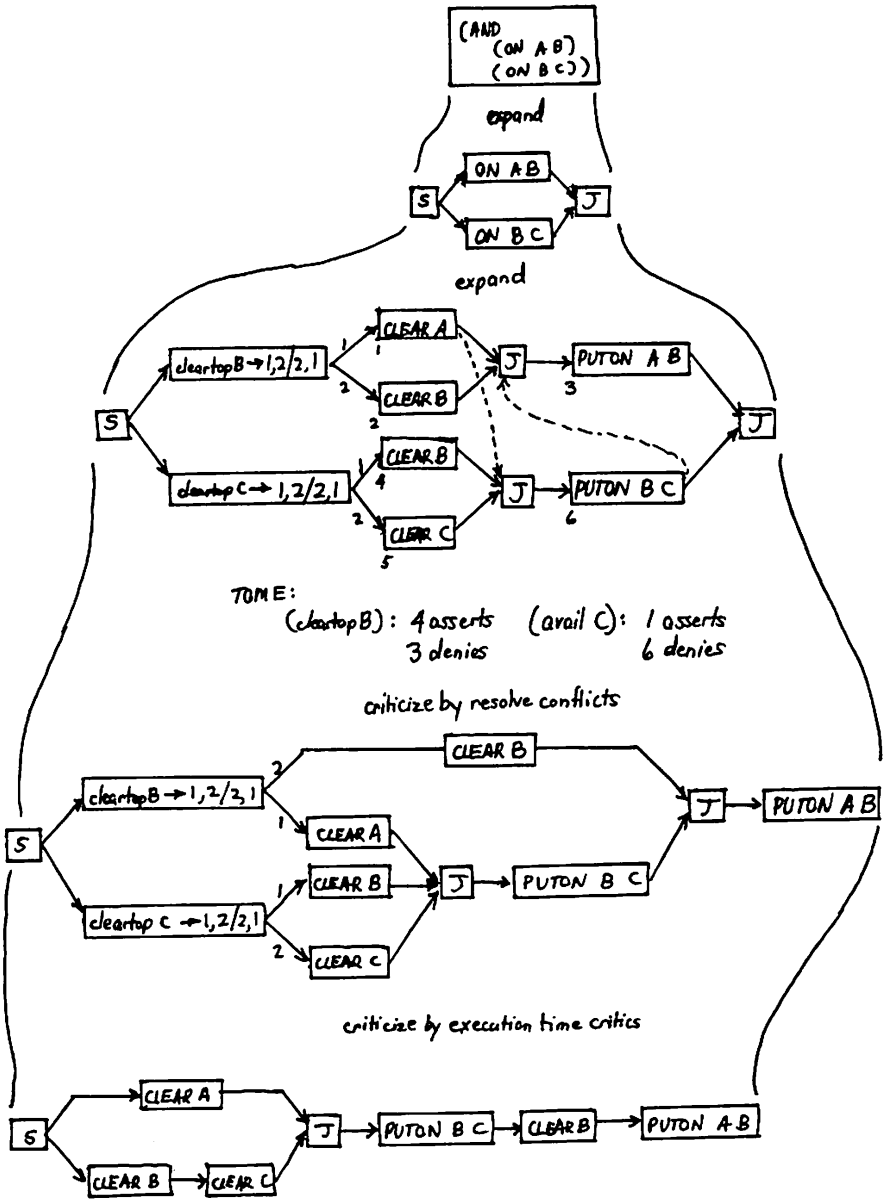


Figure 4.2: Solution for the three blocks problem

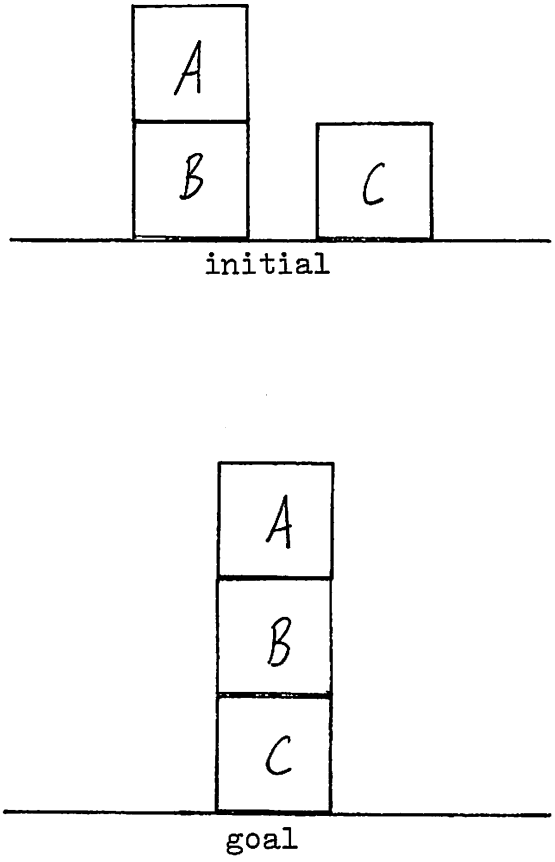


Figure 5.1: The creative destruction problem

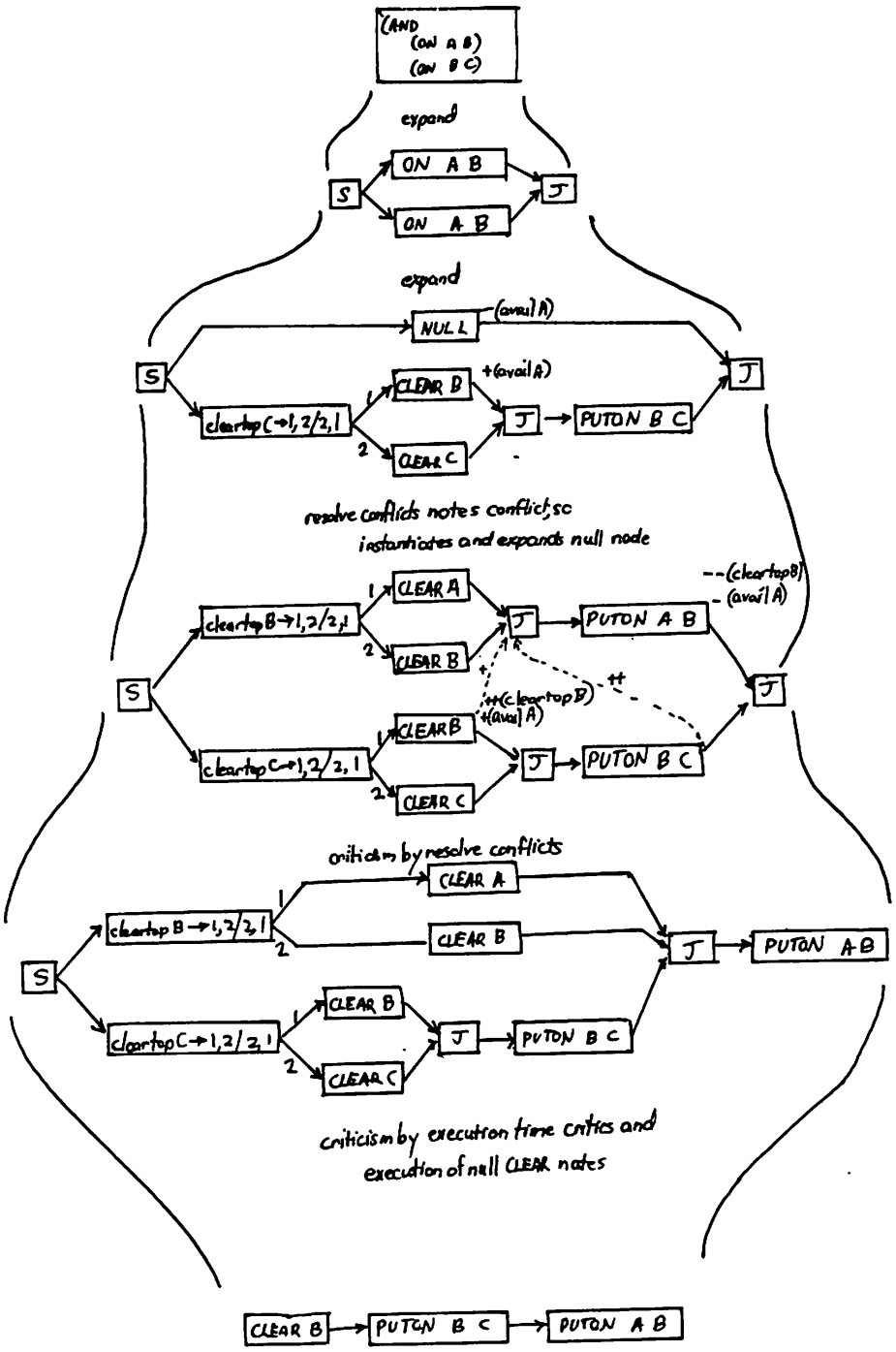


Figure 5.2: Solution for the creative destruction problem

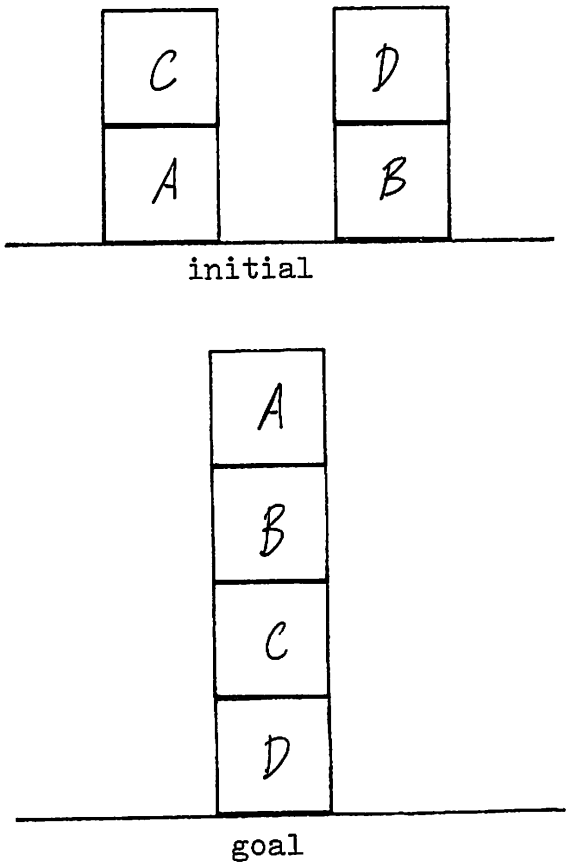


Figure 6.1: The four blocks problem

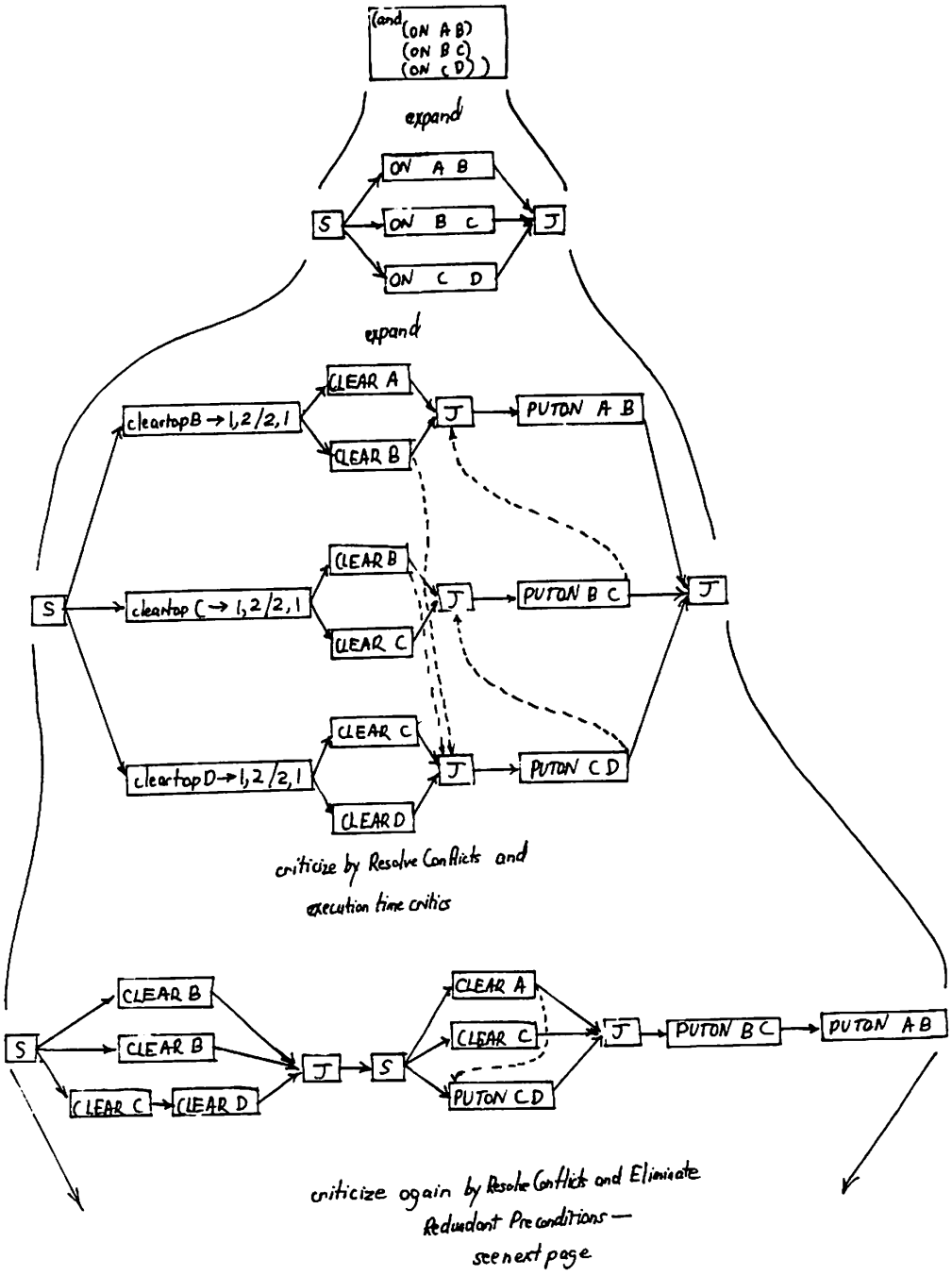


Figure 6.2: Solution for the four blocks problem

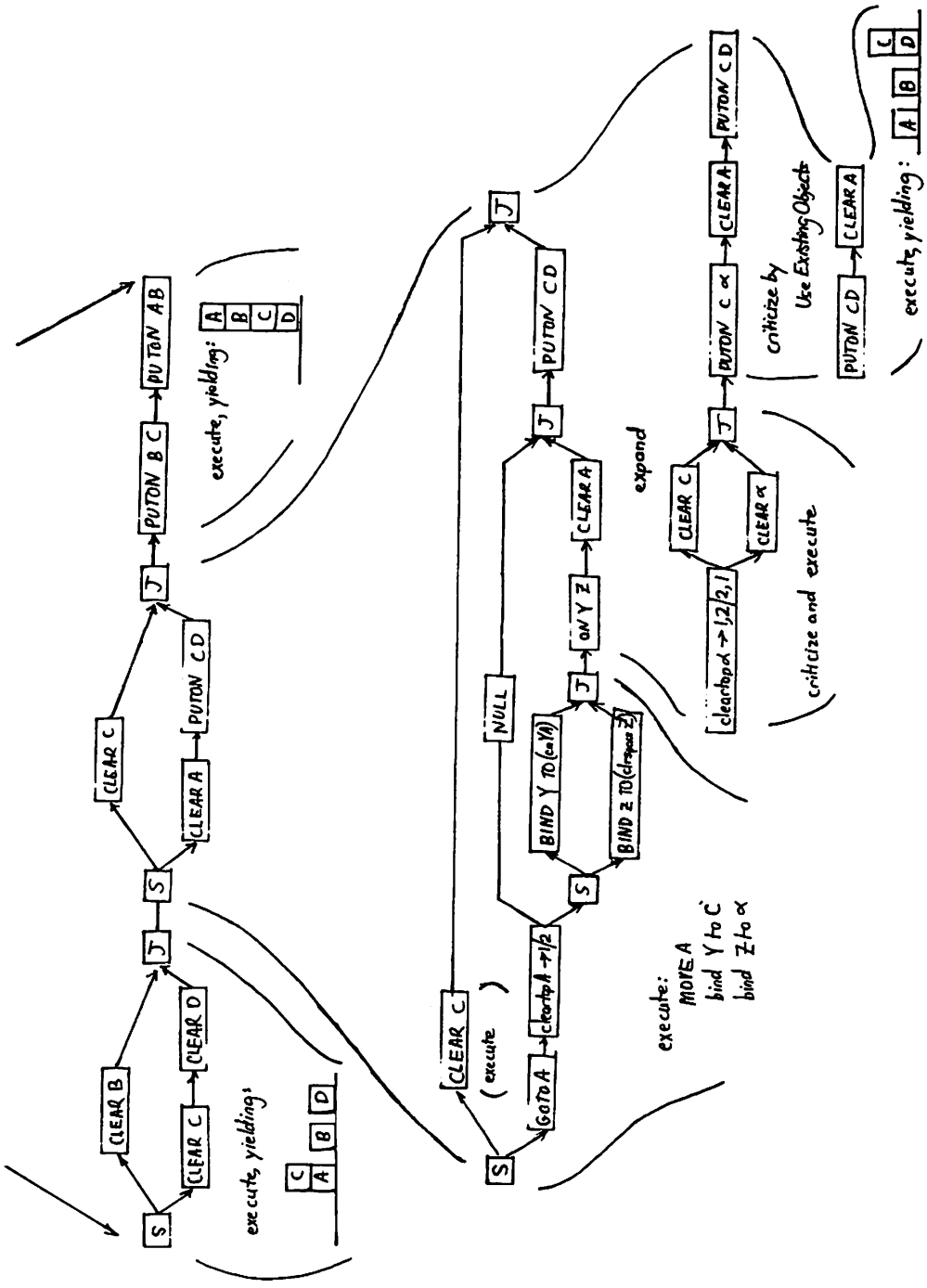
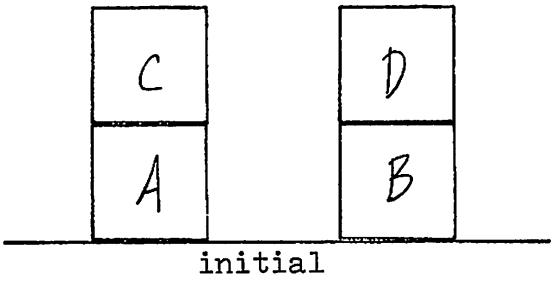


Figure 6.2: (continued)



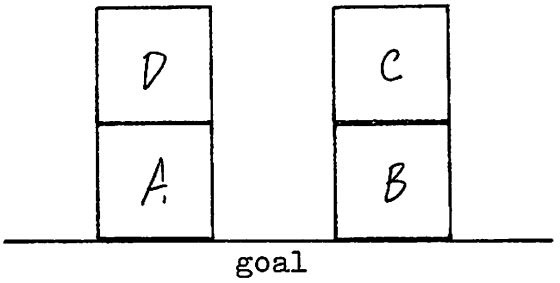


Figure 7.1: The swapping blocks problem

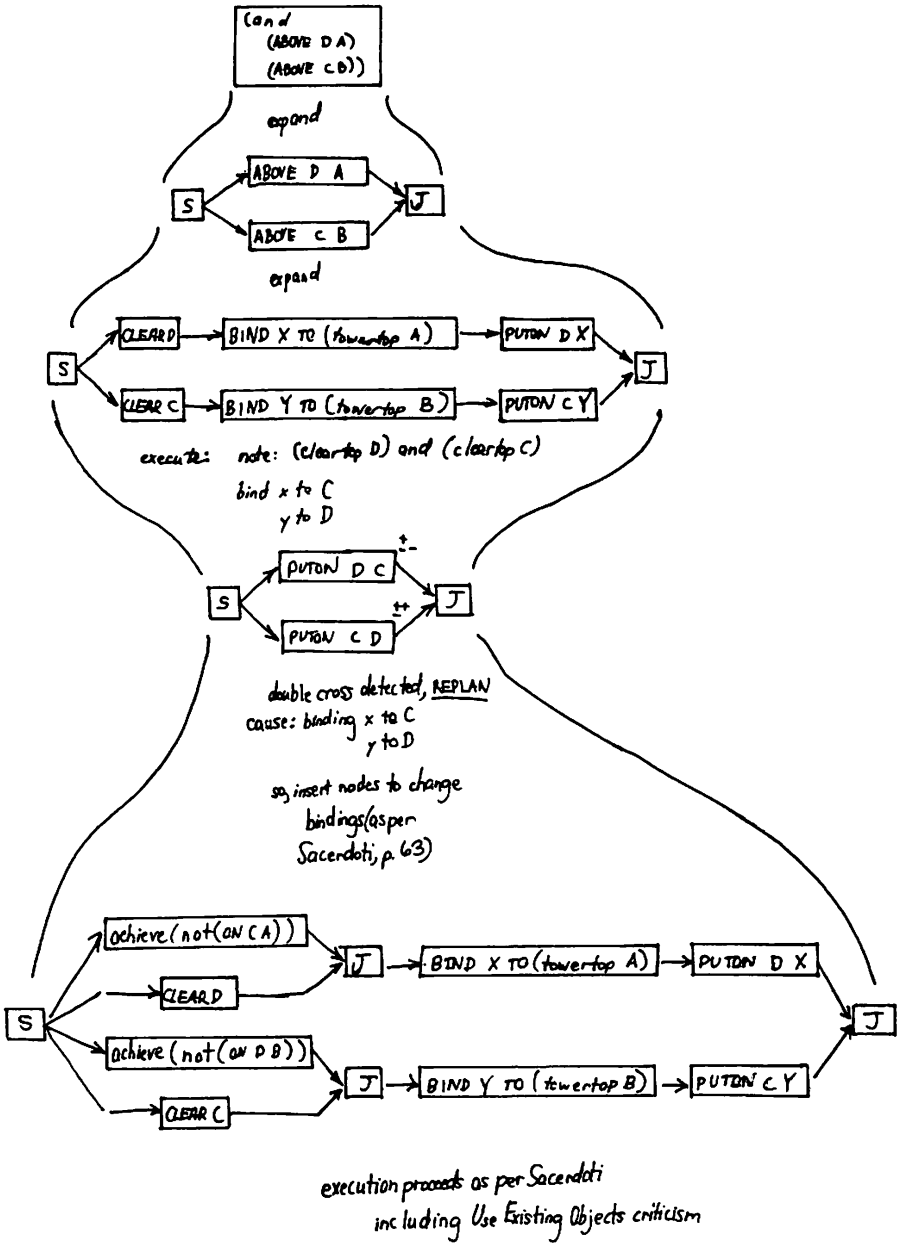
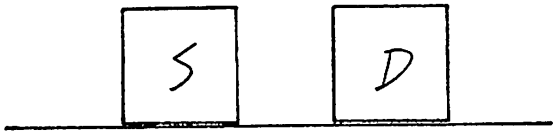
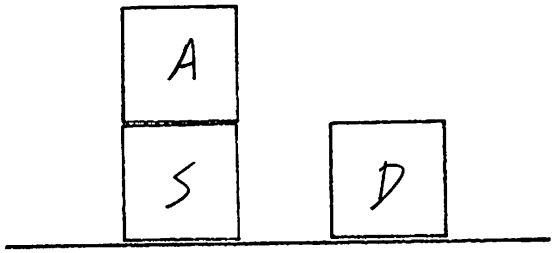


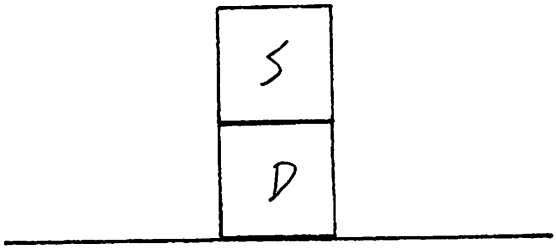
Figure 7.2: Solution for the swapping blocks problem



initial world



initial world view



goal

Figure 8.1: The missing blocks problem -- clear source thought covered

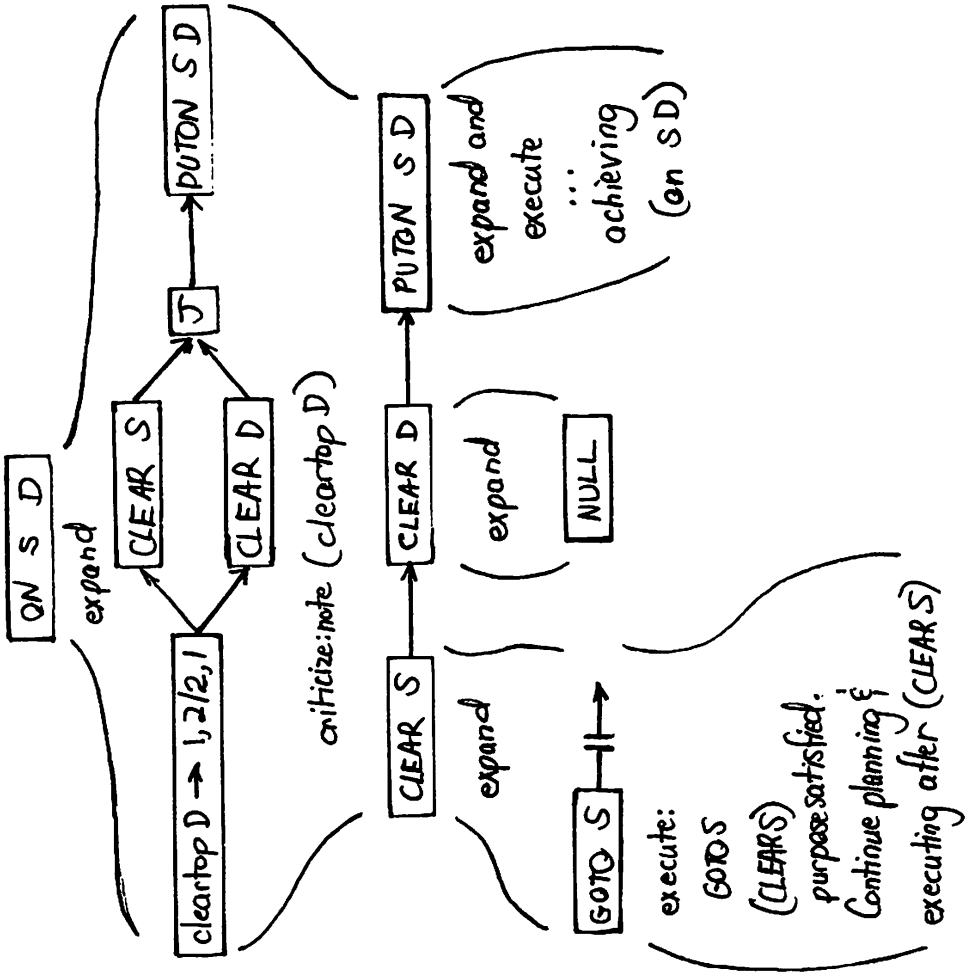
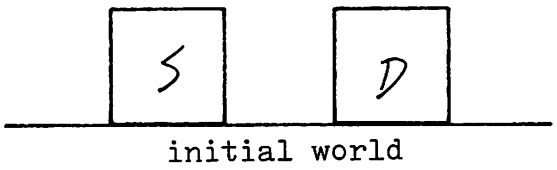
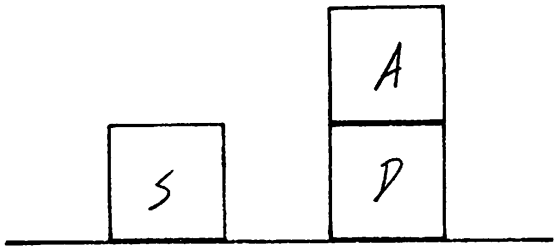


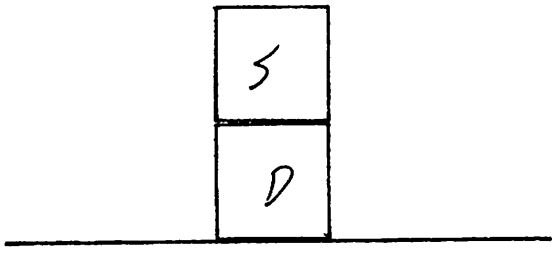
Figure 8.1: Solution for the missing blocks problem -- clear source thought covered



initial world



initial world view



goal

Figure 9.1: The missing blocks problem -- clear destination thought covered

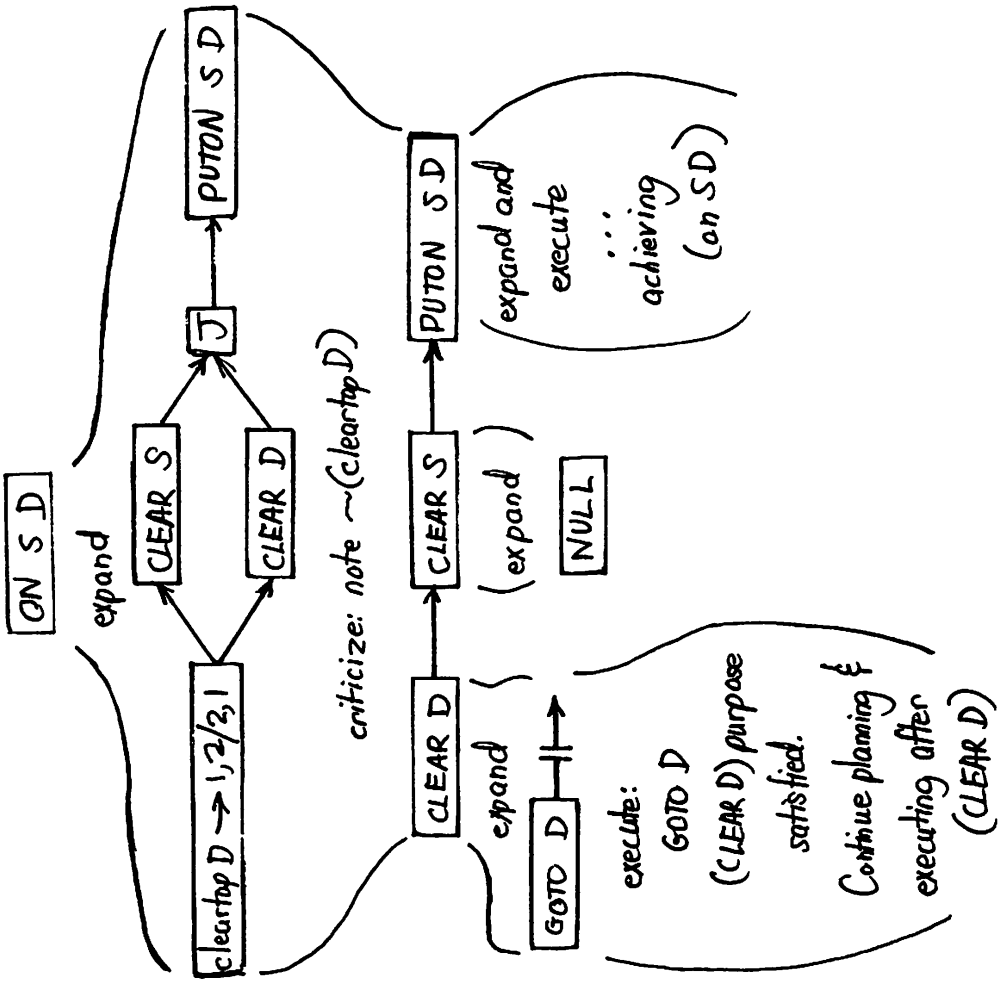
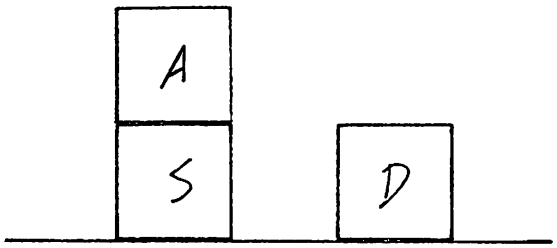
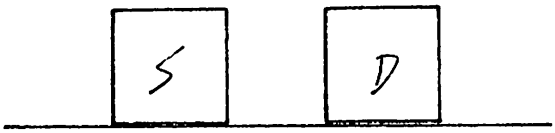


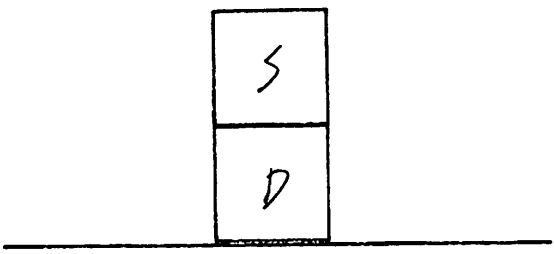
Figure 9.2: Solution for the missing blocks problem -- clear destination thought covered



initial world



initial world view



goal

Figure 10.1: The extraneous blocks problem -- covered source thought clear

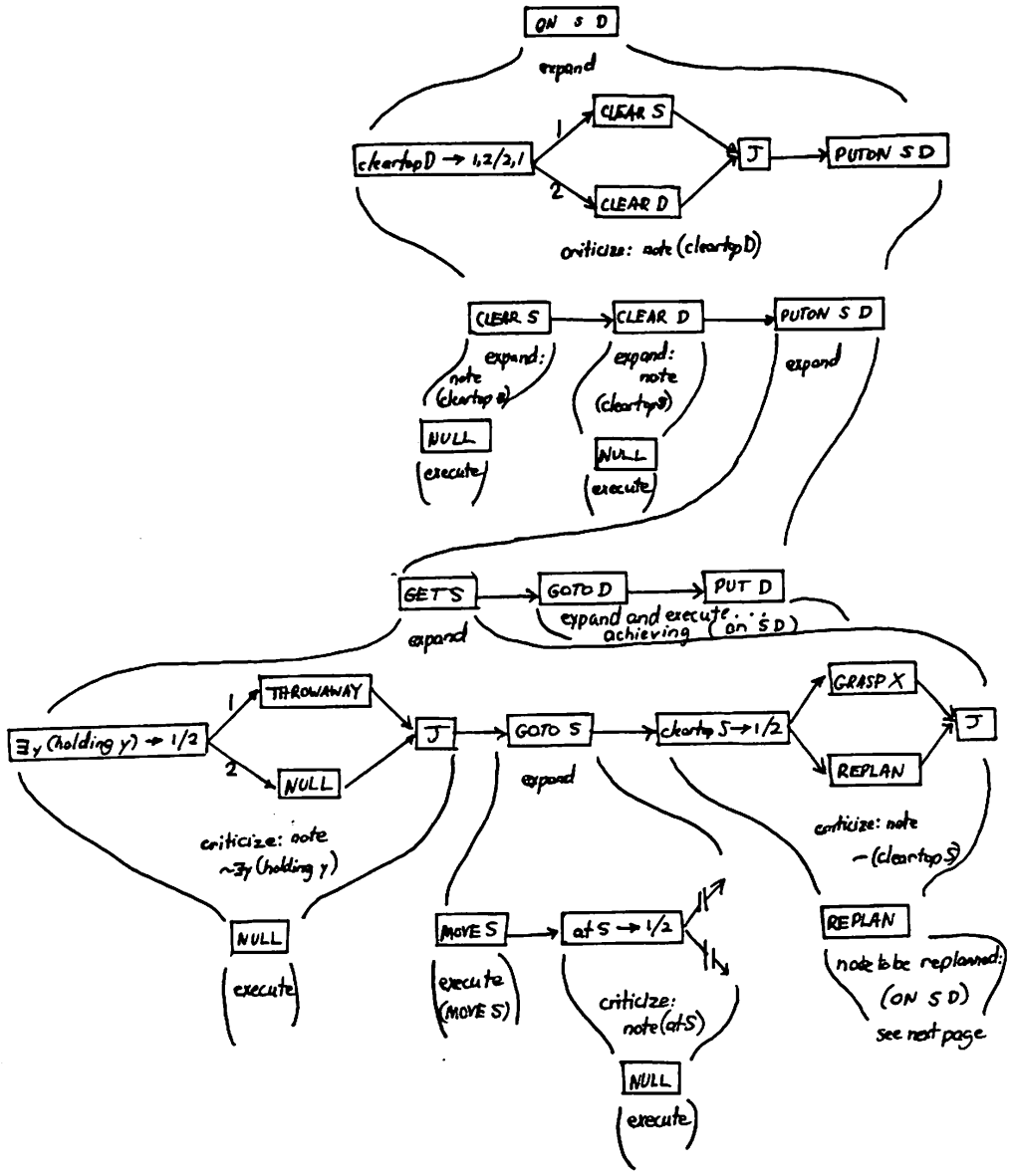


Figure 10.2: Solution for the extraneous blocks problem -- covered source thought clear

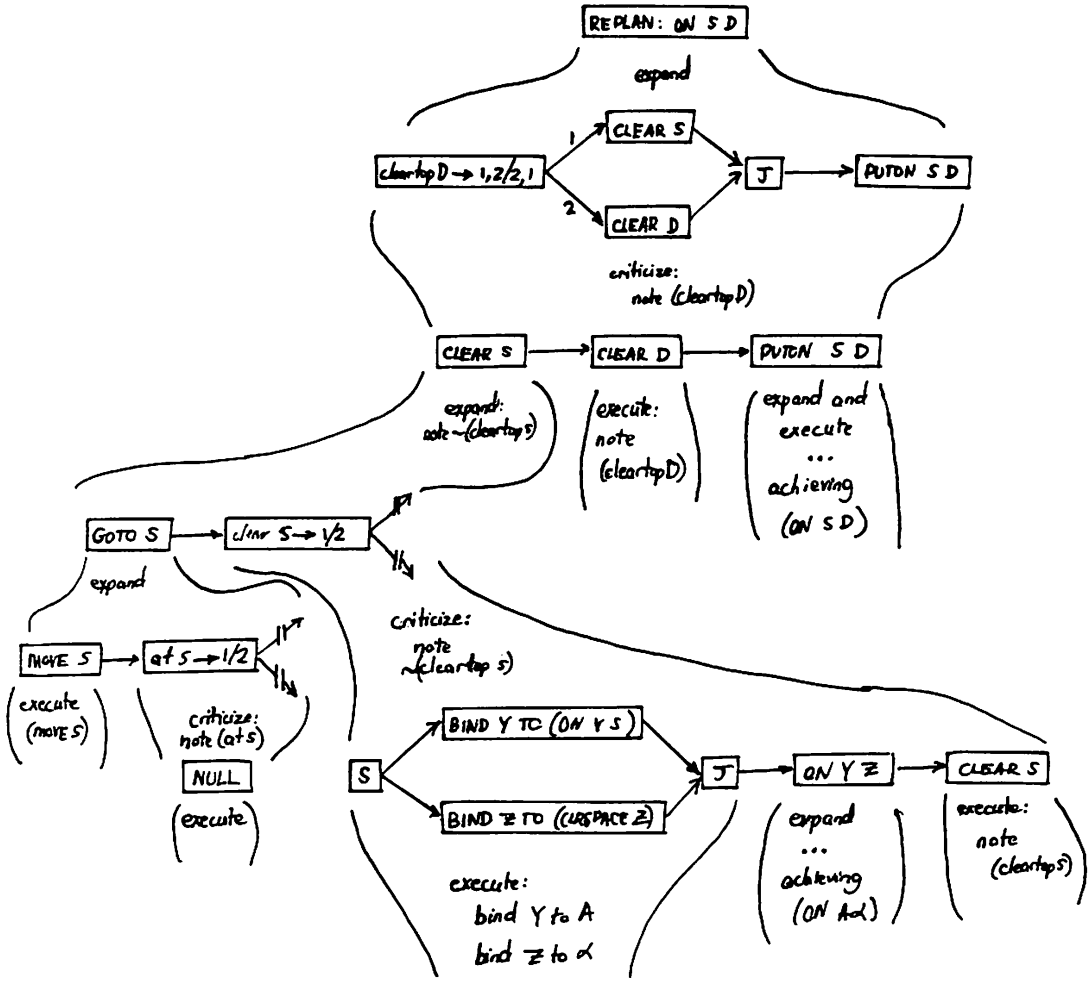
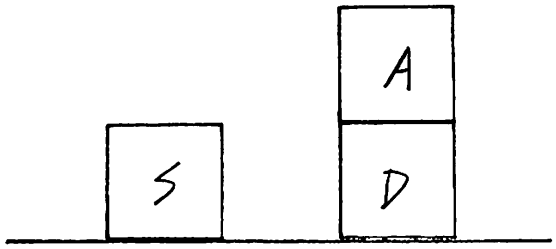
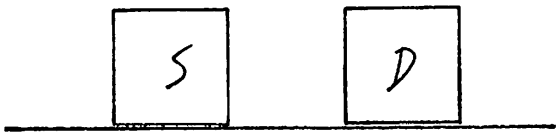


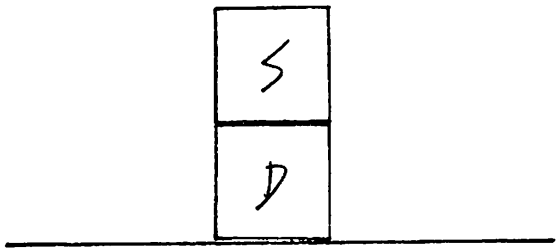
Figure 10.2: (continued)



initial world



initial world view



goal

Figure 11.1: The extraneous blocks problem -- covered destination thought clear

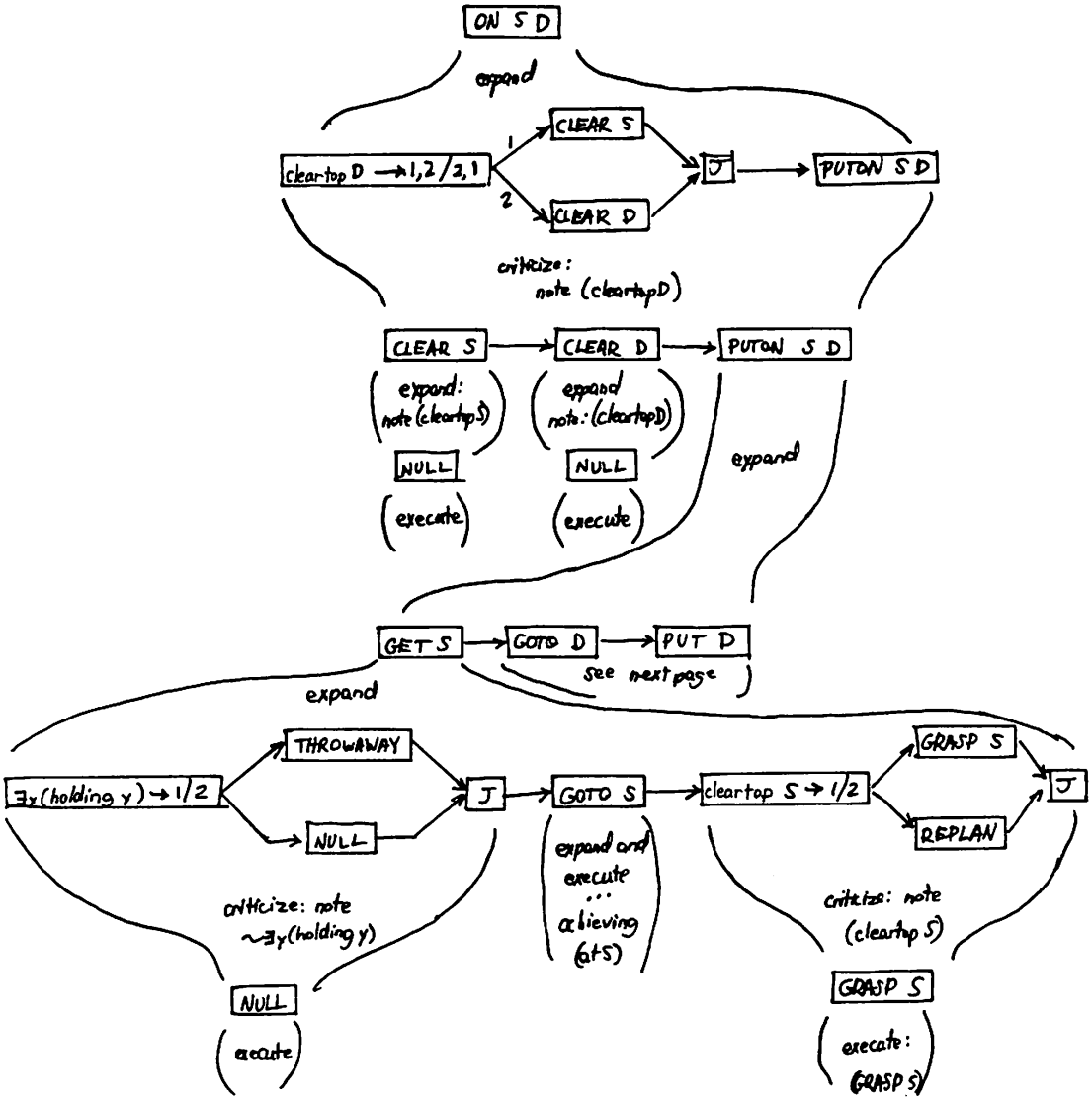


Figure 11.2: Solution for the extraneous blocks problem -- covered destination thought clear

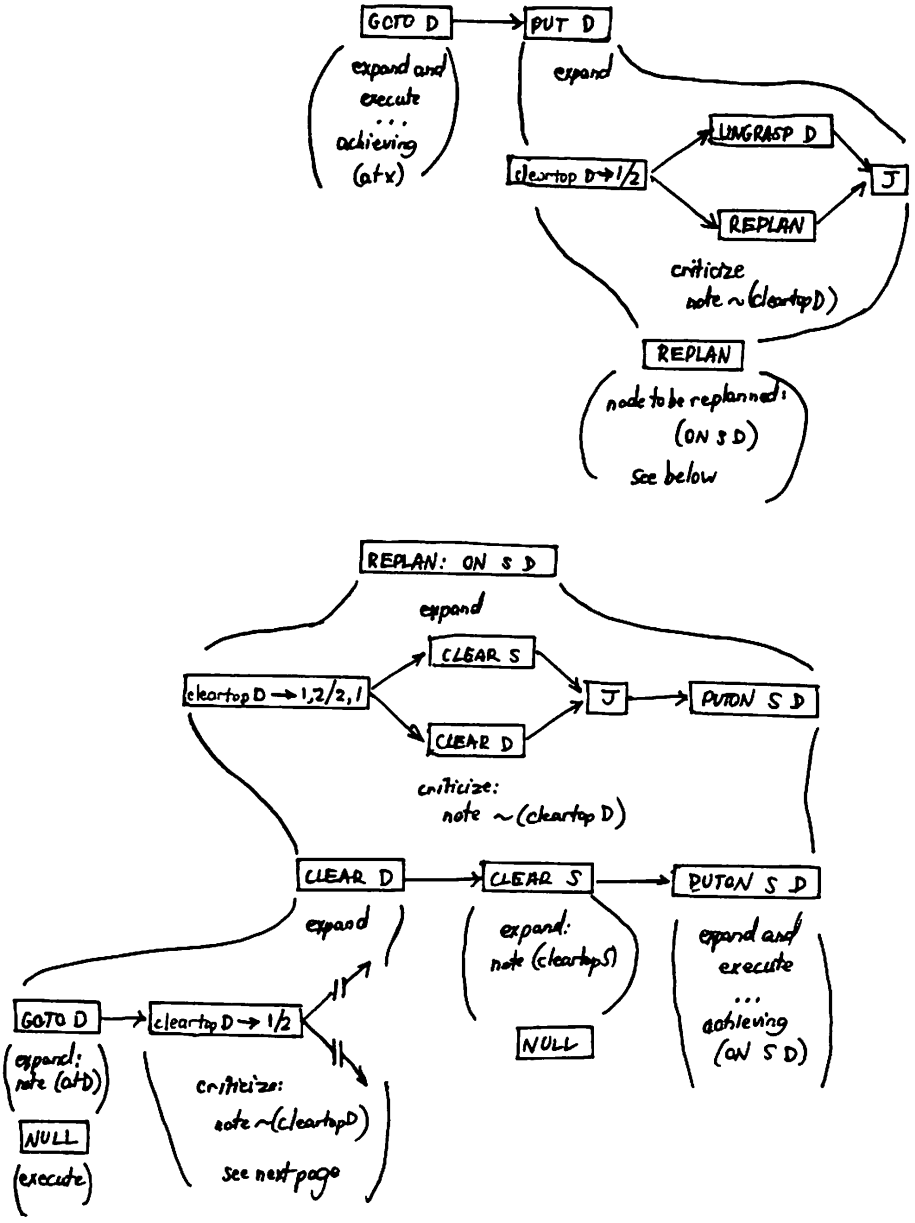


Figure 11.2: (continued)

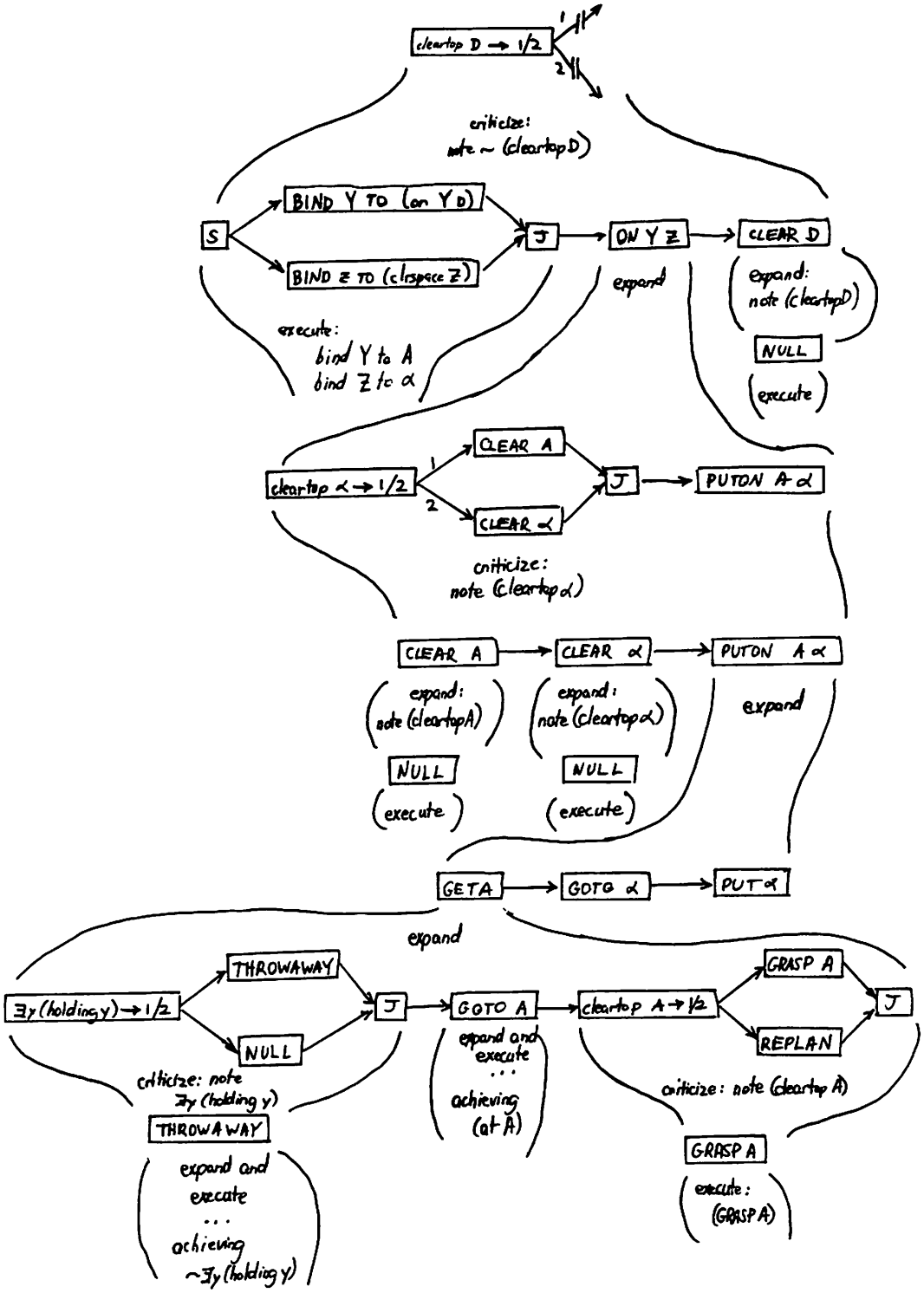
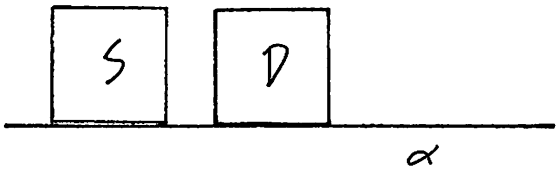
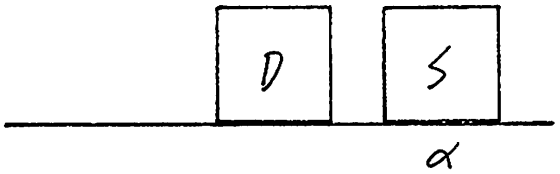


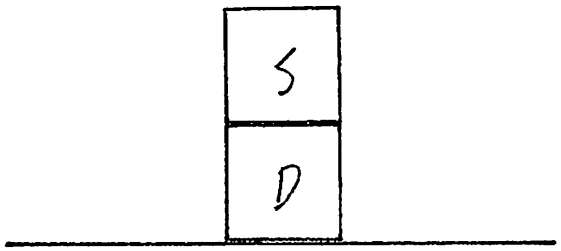
Figure 11.2: (continued)



initial world view



initial world



goal

Figure 12.1: The find block problem

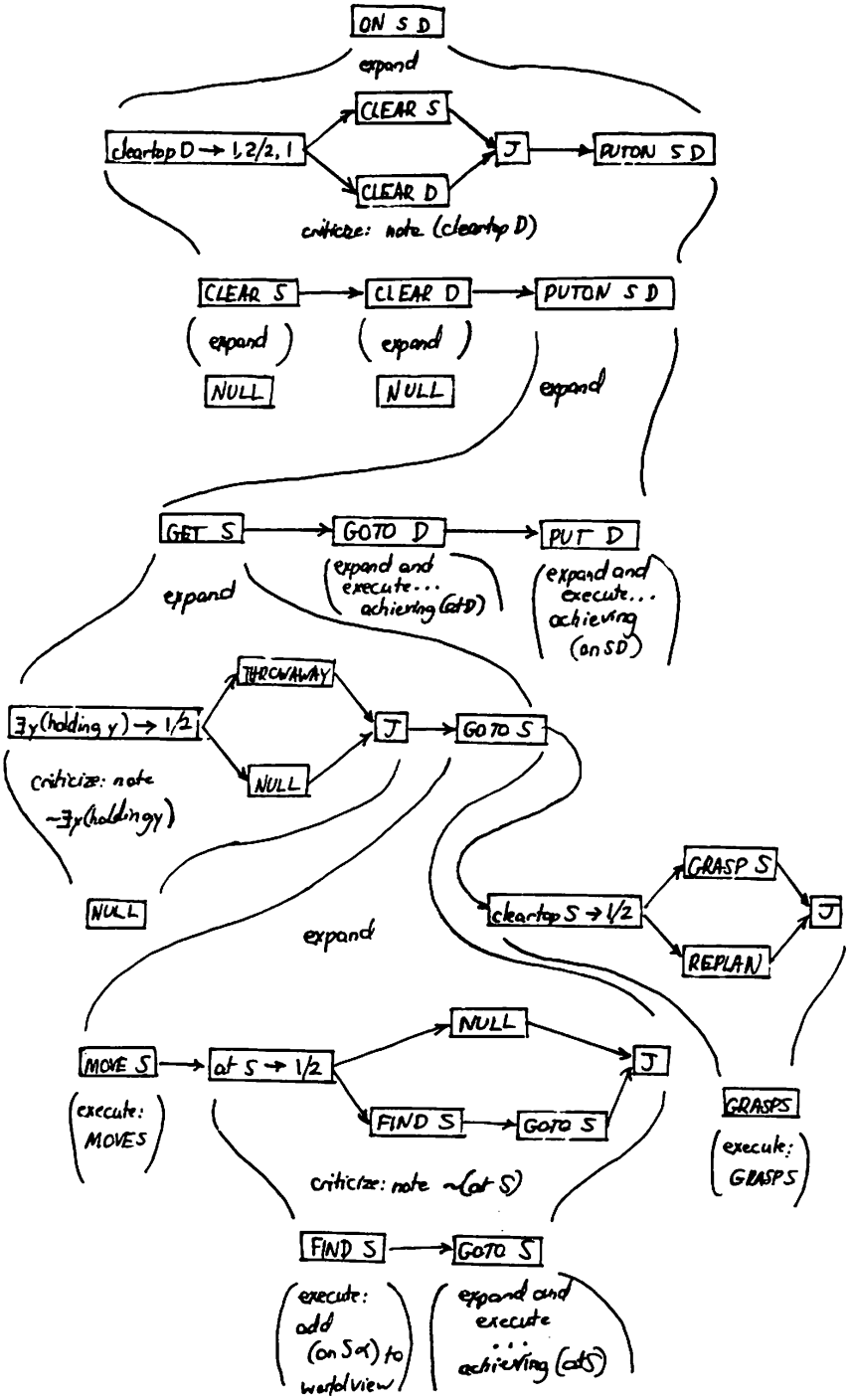


Figure 12.2: Solution to the find block problem

Appendix B: Substantiation of plan detail results

The following are arguments based on simple models of error distribution and plan structure. They are meant as informal guides to the rationale behind certain new NOAH design decisions and not as formalizations of the operation of new NOAH.

The probability of error increases exponentially with time

Let p be the probability of an error occurring during a given time unit. p is constant according to assumption 2. Let R_n be the probability of error at or before time n. Then:

$$R_n = R_{n-1} + (1-R_{n-1})p$$
$$= (1-p)R_{n-1} + p$$

so,

$$R_n = (1-p)^n R_0 + p\sum_{i=0}^{n-1}(1-p)^i$$

Now,

$$\sum_{i=0}^{n-1} x^i = \sum_{i=0}^{\infty} x^i - \sum_{i=n}^{\infty} x^i$$
$$= (1-x)^{-1} - x^n(1-x)^{-1}$$

so,

$$R_n = (1-p)^n R_0 + p\left[(p^{-1}) - (1-p)^n p^{-1}\right]$$
$$= (1-p)^n R_0 + 1 - (1-p)^n$$
$$= 1 + (R_0 - 1)(1-p)^n$$
$$= 1 - k(1-p)^n \quad \text{where } 0 \le k \le 1$$

Thus, the probability of error versus time is an exponential

with an asymptote of 1.

The size of new NOAH hierarchies is bounded linearly by the depth of expansion

We present a very simple model for procedural net expansion and prove the theorem with respect to this model.

We ignore nonlinearity and model all expansion templates as linear orderings of exactly c child nodes which may be all primitive or all non-primitive but not a combination. We model hierarchies of procedural nets as trees of nodes as follows: The nodes of the tree correspond to the nodes in the procedural nets. The arcs in the tree connect a procedural net node with the c nodes that it directly generates through one expansion. Thus, leaves of the tree correspond to primitive or unexpanded nodes. We further assume that all primitive nodes will be generated at the same depth d from the root (the root counted as depth 0).

First, we examine the number of nodes in a NOAH tree. In NOAH, all nodes are expanded to primitivity before any execution. Thus, the number of nodes generated is the number of nodes in a balanced tree of depth d with a branching factor of c, namely $c^{d+1} - 1$. Note that this is exponential in d.

Now, we examine the number of nodes in a new NOAH tree just before execution. At level 0 is the single root node. Level 1 contains the c children of the root. Level 2 con-

tains the c children of the first m nodes at level 1 (m < c), that is, cm nodes. In general, the i^{th} level (i > 1) contains the c children of each of the first m nodes at the $i-1^{th}$ level. Thus, the total number of nodes in the new NOAH tree is 1 + c + cm(d-1) which is linear in d.

The savings from replanning is thus exponential for new NOAH trees relative to NOAH trees, at least just before execution of any primitive nodes. During execution, neither the exponential nature of NOAH trees, nor the linear nature of new NOAH trees are changed. All that changes is the branching factor of the early nodes at each level in the net. New NOAH trees are still bounded by the linear number of nodes, whereas full expansion leaves NOAH trees bounded only by an exponential.

Furthermore, one would expect that the functional character of the size of the actual hierarchies of procedural nets follows these general results in spite of the vast simplifications inherent in the model. Since we are concerned only with the size of the hierarchies, the lack of non-linearity in the model need not concern us. Instead of a constant branching factor, the nodes have branching factors that vary; and the average depth of primitive nodes also varies. However, on average, these numbers stay relatively stable across the hierarchy, so that the model, in some sense, represents the "average" case. Finally, though usually the entire hierarchy is not replanned on discovering an

error, we note that each subtree of the NOAH and new NOAH nets have the same size characteristics of the tree at large.

Thus, this simple model provides some substantiation for the size of the savings during replanning of new NOAH hierarchies of procedural nets.

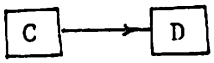
Appendix C: Problems with the Resolve Conflicts critic

The Resolve Conflicts critic has several major problems in its mode of operation within NOAH and new NOAH. Two such problems are presented here with discussion as to the general ramifications of the problems and possible methods of attack towards a solution. The severe nature of the problems make solution within the framework of the existing NOAH control structure difficult.

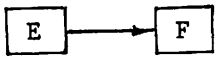
Early reordering

Implicit in the fact that NOAH criticizes and reorders after every expansion is the result that reordering of plans is done as early as possible. This mode of operation conflicts with the general principle that ordering decisions be put off as long as possible.

To see the problem explicitly, consider the following set of expansion templates: (A) expands to



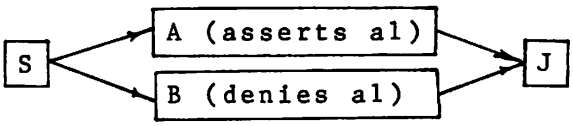
and (B) expands to



Now, suppose A asserts a1 and B denies a1. Furthermore, suppose that D asserts a1 and F denies a1. (It is quite common for the last node in an expansion to make the same asser-

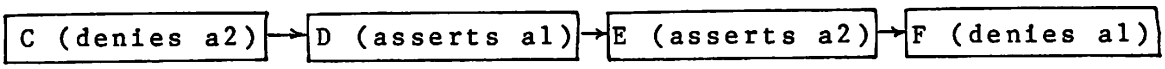
tions and denials as the parent node.) Finally, assume that C and E deny and assert respectively some independent statement a2. Such a set of expansions occurring in a real application is quite reasonable.

Given the conjunctive goal (achieve (and (A) (B))), NOAH would generate the plan:

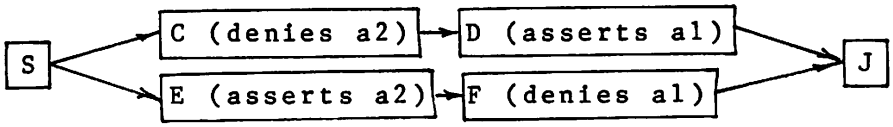


which would be reordered:

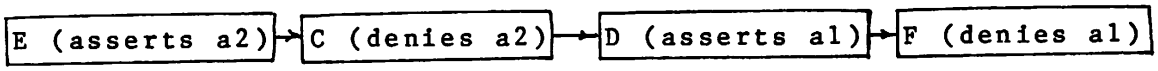
This plan would be further expanded:



But such an ordering involves a conflict between the C and E nodes. Had Resolve Conflicts waited on its initial reordering, the nonlinear plan



would be generated, which Resolve conflicts could linearize as

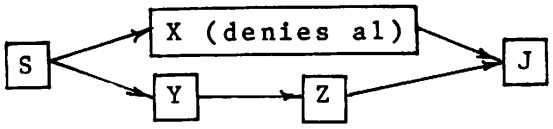


which exhibits no conflicts and is an appropriate solution. Thus, the fact that Resolve Conflicts reorders as soon as possible leaves NOAH (and new NOAH) open to errors caused by premature reordering.

Early Execution

Because new NOAH interleaves planning and execution, nodes can be executed before full reordering information is available. Thus, an agent can execute nodes that would have been postponed had planning been complete. The method of varying plan detail with time exacerbates this problem, although any system that interleaves planning and execution is prone to it.

For example, consider the plan



where X and Y are primitive, and Z expands to the single node W which asserts a1. W should be executed before X, and would if planning were to continue from this point. However, since new NOAH executes the primitive nodes at the front of the net before expanding, X and Y are executed before the W node asserting a1 is generated. Thus X and W are executed out of order.

Ordering problems in the blocks world

The small set of expansions used in NOAH's blocks world planning allows explicit checking that early reordering will not occur. Similarly, the expansions of new NOAH do not exhibit either of the problems of the previous section because specific effort was made to set up the assertions and denials that do not lead to such problems. (This was done through assertion propagation discussed below.) In general, however, not only are such efforts infeasible and undesirable for larger domains with more expansion templates, but the obvious method for rearranging the assertions and denials to prevent such problems -- the method of assertion propagation -- may not even provide a solution.

(Assertion propagation involves consistently including in the parent node of an expansion either all or none of the assertions and denials of the children nodes. In expansion template sets with loops, for instance, this method can lead to templates that still cause errors when planned in parallel.)

Approaches to general solutions to early reordering

These two problems are particularly difficult to solve in general because of their intrinsic relationship with the main control structure. Thus, solutions other than the (often infeasible) meticulous design of expansion templates tend to be drastic changes in the planning system.

One possible solution is to perform no reordering until all expansion has occurred, merely notating during expansion which nodes will involve reordering. This method allows actually putting off ordering decisions to the last possible moment and solves the early ordering problem neatly. Furthermore, the final ordering of all the nodes can be done efficiently. However, this is a major change to the NOAH control structure whose efficacy is far from ensured.

Another method might involve backtracking on discovering an incorrect linearization of nodes, more or less in the manner of the Resolve Double Cross critic. This might be implemented by assertion/denial-tracking, an extension of purpose-tracking that would check that assertions and denials were in accord.

Approaches to general solutions to early execution

Similarly, extended purpose-tracking with replanning might be an approach to the early execution problem. Purposes could be checked not only to find fortuitous situations but in general to guarantee that the nodes generated were fulfilling the purposes of higher nodes in the hierarchy. NOAH's examination of hierarchical kernels does similar checking. If a purpose were not achieved by its descendants, checking which assertion/denial conflict caused the error would allow replanning to generate a successful plan.

It should be mentioned that early execution is a function of execution without full information, which is implied by the interleaving of planning and execution. It is not an artifact of the particular control structure of the planning system, as the early reordering problem is. Quite possibly, the best solution to early execution is just to minimize the occurrence of such errors in much the same way as irreversible action errors were minimized. (See section 8.4.) Undoubtedly, solutions to these and similar problems will involve problem-solving systems of quite a different character than NOAH or new NOAH.

Appendix D: A Critique of Artificial Intelligence Research Methods

The field of artificial intelligence has in the past lost some credence among portions of the scientific community as being "vacuous and based more on slogans than achievement." [15] Nils Nilsson has noted that "several people outside of AI, whose opinions command deserved respect, are inclined to provide answers that challenge our views [of AI research]." Certainly, AI has accomplished much in its brief history, so that such disdain may be unnecessarily excessive. Yet certain aspects of AI research do give rise to deserved criticism. Thus, this (possibly overblown) rift between AI and more mainstream computer science is the result of problems on both sides.

The problems AI attempts to solve are enormous and the history of research in the field is brief at best. Thus, some typical problems of a bourgeoning young science are manifest in AI research. The field lacks good metrics for evaluation of its research projects; cross-comparison of methods has therefore been mostly anecdotal. This problem is exacerbated by the high degree of orthogonality among AI research projects due to the size of the field and the relatively small number of participating researchers. Finally, there is no standard of acceptability of results as there are in other fields, viz. rigorous proof, or statistical significance. Again, arguments for acceptability tend to be anecdotal.

The vocabulary of AI includes terms with high connotative content -- "knowledge," "world view," "belief." Consequently, terms are easily misconstrued by those unaware of their implicit meanings in the shared context among AI researchers. Possibly, more explicit definition of such terms is called for to prevent the overgeneralizations that the language tends to promote. Inevitably, as the science develops over time, this technical vocabulary will begin to be recognized as such, just as it has in pure mathematics.

Another more serious cause of overgeneralization by followers of AI research is the lack of explicit bounds set by researchers in their problem definition or solution presentation. Out of the context of such explicit limits, readers can draw conclusions that are unfounded by the research. To a certain extent, this is also the result of unfamiliarity with the shared context of AI. To an equal extent at least, it is due to oversight or disinterest on the part of researchers. In AI especially, because of its easily generalized vocabulary, explicit statements about the scope of the problems and solutions presented are imperitive.

Finally, AI has been rightly accused of a lack of formalism. To a great extent, however, AI is an engineering discipline, subject to the same rule-of-thumb methodologies as, say, chemical engineering. Furthermore, AI is a new and broad field, still with few actual systems to examine and

compare as necessary first steps toward formalization. Nonetheless, if AI has any claim towards being a science, efforts at formalization are crucial. To a certain degree, this is being done now. A case in point is Rosenschein's work on the formalization of planning systems [16] As more systems are implemented, such work will become increasingly important, and should not be neglected for lack of "practical significance."

Nilsson recommends that "AI researchers ought not to forfeit to others the tasks of defining our goals and prospects and of describing our accomplishments." Toward this end, then, AI researchers must become more explicit where they have previously been silent. The bounds and limits of their research efforts should be made concrete, as should their vocabulary. Metrics for comparison of systems and standards of acceptability for results should be developed. Formalism should begin to be pursued as the implementation of more systems makes such efforts feasible. In general, the development of "a set of goals, methodologies, principles and techniques," [15] should be an important task of AI.

Appendix E: Bibliography

[1] Appelt, Douglas E. "A Planner for Reasoning About Knowledge and Belief." Proceedings of the AAAI, August, 1980.

[2] Bledsoe, W. W. "Non-Resolution Theorem-Proving." Artificial Intelligence, volume 9, 1977.

[3] Corkill, Daniel D. "Hierarchical Planning in a Distributed Environment." Proc. of the 6th Intl. Joint Conf. on AI (IJCAI).

[4] Davis, Randall. "Report on the Workshop on Distributed AI." SIGART Newsletter, October 1980.

[5] Doyle, Jon. "Truth Maintenance." MIT AI Memo 461 February, 1978.

[6] Ernst, G. and Allan Newell. GPS: A Case Study in Generality and Problem Solving. Academic Press: New York, 1969.

[7] Fahlman, Scott E. "A Planning System for Robot Construction Tasks." Artificial Intelligence, volume 9, 1977.

[8] Fikes, Richard E. and Nils J. Nilsson. "STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving." Artificial Intelligence, volume 2, numbers 3/4, 1971.

[9] Guzman-Averas, Adolfo. Computer Recognition of Three-Dimensional Objects in a Visual Scene. MIT Project MAC. Dept. of Electrical Engineering Ph.D. Thesis. March, 1969.

[10] Konolige, Kurt. "A First Order Formalization of Knowledge and Action for a Multiagent Planning System." SRI International. Draft, April, 1980.

[11] Konolige, Kurt and Nils Nilsson. "Multiple Agent Planning Systems." Proc. of AAAI, August, 1980.

[12] Kornfeld, William A. and Carl Hewitt. "The Scientific Community Metaphor." Draft, to be submitted to IEEE Journal of Systems, Man and Cybernetics (IEEE SMC).

[13] Lesser, Victor R. and Daniel D. Corkill. "Functionally-Accurate Cooperative Distributed Systems." IEEE SMC, January, 1981.

[14] Lesser, Victor R. and Lee D. Erman. An Experiment in Distributed Interpretation. USC/Information Sciences Institute RR-79-26. May, 1979.

[15] Nilsson, Nils. "Artificial Intelligence: Engineering, Science, or Slogan?" SRI International. Draft, April, 1981.

[16] Rosenschein, Stanley J. "Plan Synthesis: A Logical Perspective." Draft, March, 1981.

[17] Rosenschein, Stanley J. private communication. March, 1981.

[18] Sacerdoti, Earl D. A Structure for Plans and Behavior. Elsevier North-Holland, Inc: New York, 1977.

[19] Sacerdoti, Earl D. "Planning in a Hierarchy of Abstraction Spaces." Artificial Intelligence, volume 5, number 2, 1974.

[20] Smith, Reid G. and Randall Davis. "Distributed Problem Solving: The Contract Net Approach." Proceedings of the Workshop on Distributed Sensor Nets. Carnegie-Mellon University, December, 1978.

[21] Smith, Reid G. and Randall Davis. "Frameworks for Cooperation in Distributed Problem Solving." IEEE SMC, January, 1981.

[22] Wilkins, Dave and Ann Robinson. "An Interactive Planning System." Submitted to IJCAI, 1981.

[23] Winograd, Terry. Understanding Natural Language. Academic Press: New York, 1972.

[24] Winston, Patrick H. Artificial Intelligence. Addison-Wesley, Inc.: Reading, Mass, 1977. Pp. 379-386.